



TUGAS AKHIR - KI091391

RANCANG BANGUN MAZE GENERATOR UNTUK ROLEPLAYING GAME “THE WARRIOR”

I Ketut Megi Trisnawan
NRP 5111100190

Dosen Pembimbing
Imam Kuswardayan, S.Kom., M.T.
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2015



FINAL PROJECT - KI091391

DESIGN AND IMPLEMENTATION OF MAZE GENERATOR FOR ROLEPLAYING GAME “THE WARRIOR”

I Ketut Megi Trisnawan
NRP 5111100190

Advisor
Imam Kuswardayan, S.Kom., M.Kom.
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

DEPARTMENT OF INFORMATICS
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

LEMBAR PENGESAHAN

Rancang Bangun Maze Generator untuk Roleplaying Game “The Warrior”

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Interaksi, Grafika dan Seni
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :

I Ketut Megi Trisnawan

NRP : 5111 100 190

Disetujui oleh Dosen Pembimbing tugas akhir :

Imam Kuswardayan, S.Kom., M.T.

NIP: 197612152003121001

(pembimbing 1)

Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

NIP: 197104281994122001

(pembimbing 2)

SURABAYA

JUNI 2015

LEMBAR PENGESAHAN

Rancang Bangun Maze Generator untuk Roleplaying Game "The Warrior"

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Interaksi, Grafika dan Seni
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :

I Ketut Megi Trisnawan

NRP : 5111 100 190

Disetujui oleh Dosen Pembimbing tugas akhir :

Imam Kuswardayan, S.Kom., M.T.

NIP: 197612152003121001

(pembimbing 1)

Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

NIP: 197104281994122001

(pembimbing 2)

SURABAYA

JUNI 2015

Rancang Bangun Maze Generator untuk Roleplaying Game “The Warrior”

Nama Mahasiswa : I Ketut Megi Trisnawan
NRP : 5111100190
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing 1 : Imam Kuswardayan, S.Kom., M.T.
Dosen Pembimbing 2 : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

ABSTRAK

Dunia game saat ini berkembang dengan pesat. Banyak game baru yang bermunculan. Berbagai genre pun diusung game tersebut. Salah satu genre yang cukup populer adalah roleplaying game (RPG).

Roleplaying game adalah genre game dimana pemain diberikan kebebasan untuk mengatur karakter dalam pengaturan fiksi. Pemain juga diberikan kebebasan untuk melakukan suatu tindakan yang akan mempengaruhi jalan cerita pada game tersebut. Dalam perkembangannya kebanyakan game bergenre roleplaying game memiliki lingkungan/tempat bermain yang statis, artinya setiap dimainkan kembali lingkungan pemain di dalam game akan sama terus. Game The Warrior mencoba menghadirkan suatu yang berbeda dengan membuat lingkungan dalam permainan berbeda setiap dimainkan.

Game The Warrior mengambil lingkungan yang berbentuk maze. Di permulaan permainan akan dijalankan sebuah algoritma yang akan membuat maze secara acak namun bisa terpecahkan. Selain itu di dalam maze akan terdapat musuh-musuh yang harus dibunuh pemain sebelum memecahkan maze tersebut.

Kata Kunci: Maze Generator, Roleplaying Game

Design And Implementation Of Maze Generator For Roleplaying Game “The Warrior”

Student Name : I Ketut Megi Trisnawan
Student ID : 5111100190
Major : Teknik Informatika FTIf-ITS
Advisor 1 : Imam Kuswardayan, S.Kom., M.T.
Advisor 2 : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

ABSTRACT

Game world is currently growing rapidly. Many new games are popping up. Various genres also carried the game. One genre that is quite popular is the roleplaying game(RPG).

Roleplaying game is a genre of game where players are given the freedom of weeks to set up character in a fictional setting. Players are also given the freedom to perform an action that will affect the storyline in the game. In the development of most games genre role-playing game has environmental/playground is static, thats mean every every time the game played again the environment will still same like the first game played. Game The Warrior try a unique way to create different environments in the game every time that game played.

Game The Warrior took environments in maze. The beginning of the game will run an algorithm that will make the maze randomly but can be solved. Also in the maze there will be enemies who must be killed before the players solve the maze.

Keywords: Maze Generator, Roleplaying Game

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan tugas akhir yang berjudul :

“Rancang Bangun Maze Generator untuk Roleplaying Game “The Warrior””

Harapan dari penulis semoga apa yang tertulis di dalam buku tugas akhir ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini, serta dapat memberikan kontribusi yang nyata.

Dalam pelaksanaan dan pembuatan tugas akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak, tanpa mengurangi rasa hormat penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Tuhan Yang Maha Esa atas segala rahmat-Nya, sehingga penulis dapat menyelesaikan Tugas Akhir.
2. Bapak penulis, I Ketut Pinti, Ibu penulis, Ni Luh Mindi, dan keluarga yang selalu memberikan dukungan penuh untuk menyelesaikan tugas akhir ini.
3. Bapak Imam Kuswardayan dan Ibu Nanik Suciati selaku dosen pembimbing yang telah bersedia meluangkan waktu untuk memberikan petunjuk selama proses pengerjaan tugas akhir ini.
4. Bapak, Ibu dosen Jurusan Teknik Informatika ITS yang telah dengan sabar mendidik dan memberikan ilmu yang tak ternilai harganya bagi penulis.
5. Seluruh staf dan karyawan Teknik Informatika ITS yang banyak memberikan kelancaran administrasi akademik kepada penulis.
6. Ibet, Melfa, Kaspul, Fandi, Ajong dan Kemal terima kasih telah mengingatkan saya akan mantan.
7. Bayu yang memperbolehkan wifinya dipakai.

8. Teman-teman user TA Lab Interaksi, Grafika dan Seni yang telah memberikan banyak dukungan dan semangat kepada penulis.
9. Teman-teman angkatan 2011 jurusan Teknik Informatika ITS yang memberikan dorongan motivasi dan bantuan kepada penulis.
10. Serta pihak-pihak lain yang tidak dapat disebutkan satu persatu.

Penulis telah berusaha sebaik-baiknya dalam menyusun tugas akhir ini, namun penulis mohon maaf apabila terdapat kekurangan, kesalahan maupun kelalaian yang telah penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, Juni 2015

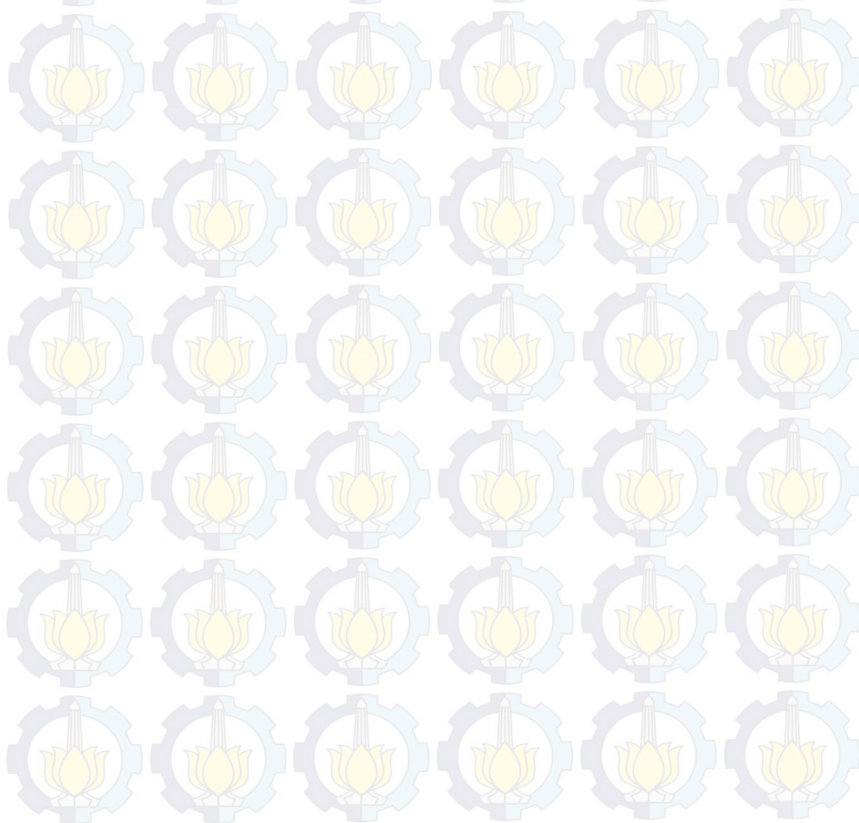
I Ketut Megi Trisnawan

DAFTAR ISI

LEMBAR PENGESAHAN	v
ABSTRAK	vii
ABSTRACT	ix
KATA PENGANTAR.....	xi
DAFTAR ISI	xiii
DAFTAR GAMBAR.....	xvii
DAFTAR TABEL	xxi
DAFTAR KODE SUMBER.....	xxiii
1 BAB I PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Tujuan.....	2
1.3. Rumusan Permasalahan.....	2
1.4. Batasan Permasalahan	2
1.5. Metodologi	2
1.6. Sistematika Penulisan	4
2 BAB II DASAR TEORI.....	7
2.1. <i>Roleplaying Game</i>	7
2.2. <i>Maze</i>	7
2.3. Unity	7
2.4. Blender	7
2.5. Algoritma <i>Growing Tree</i>	8
3 BAB III ANALISIS DAN PERANCANGAN SISTEM.....	9
3.1. Analisis Sistem	9
3.2. Perancangan Permainan.....	10
3.2.1. Deskripsi Umum Perangkat Lunak.....	10
3.2.2. Spesifikasi Kebutuhan Fungsional	11
3.2.3. Spesifikasi Kebutuhan Non-Fungsional	11
3.2.4. Karakteristik Pengguna.....	11
3.3. Perancangan Sistem.....	12
3.3.1. Kasus Penggunaan Menu Utama	12
3.3.2. Kasus Penggunaan Saat Dalam Permainan	18
3.3.3. Kontrol.....	21

3.3.4.	Antarmuka	21
3.3.5.	<i>Maze</i> Generator.....	26
3.3.6.	Level dan Skenario	33
3.4.	Design Asli dan Modifikasi <i>Growing Tree</i>	34
4	BAB IV IMPLEMENTASI.....	37
4.1.	Lingkungan Implementasi	37
4.2.	Implementasi Menu Utama	37
4.3.	Implementasi Penggunaan Saat Permainan	38
4.4.	Implementasi Kontrol	44
4.5.	Implementasi Antarmuka	45
4.5.1.	Antarmuka Menu Character Creation.....	45
4.5.2.	Antarmuka Character Window	45
4.5.3.	Antarmuka Inventory Window	48
4.5.4.	Antarmuka Loot Window	49
4.5.5.	Antarmuka Shop Window	49
4.5.6.	Antarmuka Dalam Pertarungan	50
4.6.	Implementasi <i>Maze</i> Generator.....	50
4.7.	Implementasi Level dan Skenario	52
4.8.	Pembuatan Karakter	53
5	BAB V PENGUJIAN DAN EVALUASI	57
5.1.	Lingkungan Uji Coba	57
5.2.	Skenario Uji Coba	57
5.3.	Hasil Uji Coba	59
5.3.1.	Menggunakan <i>Combo</i>	59
5.3.2.	Mengurangi Darah Musuh.....	60
5.3.3.	Mendapatkan <i>Gold</i>	60
5.3.4.	Pemain Mati.....	62
5.3.5.	<i>Pause</i>	62
5.3.6.	Memunculkan Character Window.....	63
5.3.7.	Memunculkan Inventory Window.....	64
5.3.8.	Membuka Peti.....	64
5.3.9.	Mengambil <i>Item</i>	65
5.3.10.	Membeli <i>Item</i>	66
5.3.11.	Pembuatan <i>Maze</i>	66
5.4.	Evaluasi	71

5.5. Kuisisioner	71
6 BAB VI KESIMPULAN DAN SARAN.....	73
6.1. Kesimpulan.....	73
6.2. Saran.....	73
Daftar Pustaka	75
Lampiran Potongan Kode.....	77
Lampiran Kuisisioner.....	85
BIODATA PENULIS.....	90



DAFTAR TABEL

Tabel 3.1. Karakteristik Pengguna	12
Tabel 3.2. Kode Kasus Penggunaan Menu Utama	13
Tabel 3.3. Melanjutkan Permainan	13
Tabel 3.4. Memulai Permainan Baru	14
Tabel 3.5. Melihat Kontrol	14
Tabel 3.6. Menutup Perangkat Lunak	15
Tabel 3.7. Level dan Skenario	34
Tabel 4.1. Lingkungan Implementasi	37
Tabel 5.1. Lingkungan Uji Coba	57
Tabel 5.2. Tabel Skenario Uji Coba	58
Tabel 5.3. Hasil Kuisisioner	72

DAFTAR GAMBAR

Gambar 1 Diagram kasus Penggunaan Menu Utama	12
Gambar 2. Diagram Aktifitas Melanjutkan Permainan	15
Gambar 3. Diagram Aktifitas Memulai Permainan Baru	16
Gambar 4. Diagram Aktifitas Melihat Kontrol	17
Gambar 5. Diagram Aktifitas Menutup Perangkat Lunak	17
Gambar 6. FSM Pemain Menyerang	18
Gambar 7. FSM Membuka Peti.....	19
Gambar 8. FSM Character Window	19
Gambar 9. FSM Berbelanja	20
Gambar 10. FSM Keluar dari Maze	20
Gambar 11. Perancangan Menu Utama	22
Gambar 12. Perancangan Menu Creation.....	22
Gambar 13. Perancangan Character Window.....	23
Gambar 14. Perancangan Inventory Window.....	23
Gambar 15. Perancangan Loot Window.....	24
Gambar 16. Perancangan Antarmuka Shop Window	24
Gambar 17. Perancangan Antarmuka Dalam Pertarungan	25
Gambar 18. Kumpulan Sel Kubus.....	26
Gambar 19 Langkah 1 Algoritma Pada Sel Kubus 4x4.....	26
Gambar 20 Langkah 2 Algoritma Pada Sel Kubus 4x4.....	27
Gambar 21 Langkah 3 Algoritma Pada Sel Kubus 4x4.....	27
Gambar 22 Langkah 4 Algoritma Pada Sel Kubus 4x4.....	27
Gambar 23 Langkah 5 Algoritma Pada Sel Kubus 4x4.....	28
Gambar 24 Langkah 6 Algoritma Pada Sel Kubus 4x4.....	28
Gambar 25 Langkah 7 Algoritma Pada Sel Kubus 4x4.....	28
Gambar 26 Langkah 8 Algoritma Pada Sel Kubus 4x4.....	28
Gambar 27 Langkah 9 Algoritma Pada Sel Kubus 4x4.....	29
Gambar 28 Langkah 10 Algoritma Pada Sel Kubus 4x4.....	29
Gambar 29 Langkah 11 Algoritma Pada Sel Kubus 4x4.....	29
Gambar 30 Langkah 11 Algoritma Pada Sel Kubus 4x4.....	29
Gambar 31 Langkah 12 Algoritma Pada Sel Kubus 4x4.....	30
Gambar 32 Langkah 13 Algoritma Pada Sel Kubus 4x4.....	30

Gambar 33 Langkah 14 Algoritma Pada Sel Kubus 4x4.....	30
Gambar 34 Langkah 15 Algoritma Pada Sel Kubus 4x4.....	30
Gambar 35 Langkah 16 Algoritma Pada Sel Kubus 4x4.....	31
Gambar 36 Langkah 17 Algoritma Pada Sel Kubus 4x4.....	31
Gambar 37 Hasil Maze Dari Algoritma	31
Gambar 38 Maze 5x5	32
Gambar 39 Maze 6x6	32
Gambar 40 Maze 15x15	33
Gambar 41. Sel Pada Design Asli	35
Gambar 42. Semua Sel Pada Design Modifikas	35
Gambar 43. Tampilan Menu Utam	37
Gambar 44. Tampilan Character Creation.....	45
Gambar 45. Tampilan Character Window Tab Equipment	46
Gambar 46. Tampilan Tab Attributes Pada Character Window	46
Gambar 47. Tampilan Point Attribute Jika Tidak Nol.....	47
Gambar 48. Tampilan Tab Sklills Pada Character Window.....	47
Gambar 49. Tampilan Jika Point Skill Tidak Nol	48
Gambar 50. Tampilan Inventory Window.....	48
Gambar 51. Tampilan Loot Window.....	49
Gambar 52. Tampilan Shop Window	49
Gambar 53. Tampilan Dalam Pertarungan	50
Gambar 54. Screenshoot dari Unity Untuk Level 1-11	52
Gambar 55. Implementasi Level 12-20	53
Gambar 56. Karakter Utama.....	54
Gambar 57. Peti	54
Gambar 58. Monster	55
Gambar 59. All Item	55
Gambar 60. Semua Serangan <i>Combo</i>	59
Gambar 61. Darah Musuh Berkurang.....	60
Gambar 62. Gold Pemain 0	61
Gambar 63. Gold Pemain Bertambah.....	61
Gambar 64. Pemain Mati.....	62
Gambar 65. Pause	63
Gambar 66. Character Window	63
Gambar 67. Inventory Window	64

Gambar 68. Membuka Peti.....	65
Gambar 69. Mengambil <i>Item</i> dari Loot Window	65
Gambar 70. Membeli Item	66
Gambar 71. Hasil 1 Maze 6x6.....	67
Gambar 72. Hasil 2 Maze 6x6.....	67
Gambar 73. Hasil 3 Maze 6x6.....	68
Gambar 74. Hasil 1 Maze 10x10.....	68
Gambar 75. Hasil 2 Maze 10x10.....	69
Gambar 76. Hasil 3 Maze 10x10.....	69
Gambar 77. Hasil 1 Maze 14x14.....	70
Gambar 78. Hasil 2 Maze 14x14.....	70
Gambar 79. Hasil 3 Maze 14x14.....	71

BIODATA PENULIS



Penulis, I Ketut Megi Trisnawan, lahir di kota Amlapura pada tanggal 29 Mei 1992. Penulis adalah anak keempat dari empat bersaudara dan dibesarkan di kota Amlapura, Bali.

Penulis menempuh pendidikan formal di SDN 1 Karangasem(1999-2005), SMPN 2 Amlapura (2005-2008), SMAN 2 Amlapura (2008-2011). Pada tahun 2011, penulis memulai pendidikan S1 jurusan Teknik Informatika Fakultas Teknologi Informasi di Institut Teknologi Sepuluh Nopember Surabaya, Jawa Timur.

Di jurusan Teknik Informatika, penulis mengambil bidang minat Interaksi, Grafika dan Seni dan memiliki ketertarikan di bidang Pembuatan Game. Walaupun penulis belum pernah mengukir prestasi untuk kampus tercinta bukan berarti penulis tidak pernah berusaha. Penulis sudah mencoba dengan segala kemampuan namun apadaya keberuntungan tidak menghampiri penulis. Dan penulis pernah menjadi asisten Animasi Komputer dan Pemodelan 3D. Penulis dapat dihubungi melalui alamat email ketut.3snawan@gmail.com.

BAB I

PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan tugas akhir, dan sistematika penulisan.

1.1. Latar Belakang

Saat ini *video game* berkembang dengan pesat. Berbagai ide dituangkan menjadi sebuah *video game* yang kreatif dengan berbagai *genre* yang diusungnya. *Video game* menjadi suatu lahan industri yang sangat menjanjikan di masa depan. Bahkan saat ini banyak orang yang sehari-harinya hanya bermain *video game* namun bisa menghasilkan uang yang cukup banyak.

Berhasilnya suatu *video game* dipengaruhi berbagai faktor. Salah satunya lingkungan/*environment* yang menjadi tempat bermain karakter pemain. Sering kali pemain merasa cepat bosan karena lingkungan *video game* hanya itu-itu saja. Setiap berganti level keadaan lingkungan *video game* tidak berubah. Untuk membuat sebuah lingkungan yang berubah-ubah setiap level secara manual tentu memakan waktu yang cukup lama. Maka diperlukan pembuat lingkungan secara otomatis. Berbagai tema untuk lingkungan bisa dibuat dinamis, dan di tugas akhir ini tema yang akan diusung adalah *maze*.

Faktor lainnya adalah *genre* yang diusung oleh *video game* tersebut. Di dunia *video game* mengenal berbagai macam *genre* yang memiliki peminatnya sendiri-sendiri. Salah satu *genre* yang cukup banyak peminatnya adalah *roleplaying game*. Di dalam *genre* ini pemain diberikan kebebasan untuk mengatur karakter dalam pengaturan fiksi.

Tugas akhir ini akan mencoba memadukan salah satu *genre* yaitu *roleplaying game* dengan lingkungan *maze* yang dinamis, dimana setiap dimainkan *maze* akan berubah. Dengan

dinamisnya lingkungan ini diharapkan pemain tidak akan cepat merasa bosan.

1.2. Tujuan

Tujuan dari pembuatan tugas akhir ini adalah menerapkan algoritma *growing tree* untuk membuat *maze* generator pada game The Warrior dengan *genre roleplaying game*.

1.3. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana merancang dan membuat *scenario*, level dan *gameplay roleplaying game* pada *video game* The Warriror?
2. Bagaimana membuat rancangan *maze* yang sesuai dengan aturan permainan dan memungkinkan untuk diselesaikan?
3. Bagaimana membuat dan menerapkan algoritma *growing tree* untuk membuat *maze* generator pada game The Warrior?

1.4. Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Pembuatan *video game* menggunakan Unity dan bahasa pemrograman C#.
2. *Maze* yang akan dibuat termasuk visualisasi dan *scenario*.
3. Dalam permulaan permainan *maze* akan di *generate* (*maze* akan di *generate* ulang apabila pemain memilih New Game pada *menu*).

1.5. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan tugas akhir ini yaitu:

1. Studi literatur

Pada tahap ini merupakan tahap pengumpulan dan pembelajaran informasi yang akan digunakan untuk

mengimplementasikan Tugas Akhir. Literatur yang digunakan adalah sebagai berikut:

- a. *Roleplaying game*;
- b. Unity;
- c. Maze;
- d. Algoritma *Growing Tree*;

2. Analisis dan Perancangan Sistem

Pada tahap ini, akan dilakukan analisa, perancangan dan pendefinisian kebutuhan sistem untuk mengetahui permasalahan yang akan dihadapi pada tahap implementasi. Kemudian, akan dijabarkan kebutuhan-kebutuhan tersebut kedalam perancangan fitur sistem. Berikut langkah yang akan dilakukan perancangan proses aplikasi;

- a. Perancangan *maze generator*;
- b. Perancangan *gameplay*;
- c. Perancangan data dan *asset game*;
- d. Perancangan menu.

3. Implementasi

Pada tahap ini dilakukan pembuatan aplikasi permainan beserta sistem yang terkait sesuai dengan tahap sebelumnya. Aplikasi akan dibuat dengan bantuan *game engine* Unity dengan menggunakan bahasa pemrograman C#.

4. Pengujian dan evaluasi

Pada tahap ini akan dilakukan pengujian terhadap perangkat lunak menggunakan beberapa metode yaitu pengujian *blackbox*. Pengujian *blackbox* adalah metode pengujian yang berfokus pada fungsionalitas tanpa melihat internal kode untuk mengetahui kemungkinan *bug* dan error. Pengujian dilakukan dengan berbagai test case yang memungkinkan terjadinya error dan *bug*.

5. Penyusunan buku tugas akhir

Pada tahap ini dilakukan penyusunan laporan dari seluruh konsep, dasar teori, implementasi, proses yang telah dilakukan, dan hasil-hasil yang telah didapatkan selama pengerjaan tugas akhir.

1.6. Sistematika Penulisan

Buku tugas akhir ini bertujuan untuk memberikan gambaran dan dokumentasi dari pengerjaan tugas akhir yang dilakukan. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut.

Secara garis besar, buku tugas akhir ini terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan tugas akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan tugas akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai analisis perangkat lunak meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifikasi pengguna. Kemudian perancangan sistem meliputi perancangan *maze* generator, *gameplay*, data beserta *asset*, dan antarmuka.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan perangkat lunak.

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian terhadap perangkat lunak

yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi mengetahui kemampuan perangkat lunak.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan dan saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan tugas akhir.

Lampiran

Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini.

BAB II

DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan tugas akhir ini. Teori-teori tersebut meliputi .

2.1. *Roleplaying Game*

Roleplaying game merupakan *genre game* dimana pemainnya bisa mengatur karakter dalam pengaturan fiksi. Seiring dengan berjalannya waktu pengertian *roleplaying game* tidak terbatas hanya pada seperangkat pengaturan fiksi, namun juga cerita di dalamnya, sistem pertarungannya dan sistem pertumbuhan karakter pemain yang memungkinkan karakter pemain bertambah kuat seiring dengan permainan tersebut dimainkan [1].

2.2. *Maze*

Banyak orang mengartikan *maze* adalah tempat dimana terdapat dinding tinggi atau tepian, dimana kita bisa berkeliling dan tersesat [2]. *Maze* juga berarti struktur yang kompleks dengan serangkaian jalur interkoneksi. Istilah ini juga digunakan untuk merujuk kepada teka-teki grafis yang mereplikasi *maze* pada media dua dimensi. Untuk memecahkan teka-teki ini pemain harus mencari jalan ke pintu keluar atau lokasi lain [3].

2.3. *Unity*

Unity adalah sebuah platform pengembangan yang fleksibel dan kuat untuk menciptakan 3D dan 2D game multiplatform dan pengalaman interaktif. Unity merupakan platform yang lengkap bagi siapa saja yang bertujuan untuk membangun bisnis perangkat lunak yang berkualitas dan menghubungkan para pemain [4].

2.4. *Blender*

Blender adalah sebuah perangkat lunak yang gratis dan *open source* untuk animasi 3D. Mendukung keseluruhan dari 3D,

pipeline—modeling, rigging, animation, simulation, rendering, compositing dan motion tracking, bahkan video editing dan game creation. Blender adalah cross-platform dan berjalan dengan baik pada Linux, Windows dan Macintosh computer [5].

2.5. Algoritma Growing Tree

Tree merupakan salah satu contoh dari data struktur yang sering digunakan pada *computer science*. Data struktur ini memiliki *root*, *branches* dan *leaves*. Dalam penggunaannya terdapat beberapa cara antara lain *breadth first traversal* dan *depth frist traversal*. Algoritma yang akan digunakan pada tugas akhir ini yaitu *growing tree* menerapkan *depth frist traversal*.

Pada algoritma ini pertama-tama dibangun kumpulan sel kubus(tanpa bagian atas). Lalu dilakukan algoritma seperti di bawah ini:

1. Pilih secara acak satu sel, dan letakkan pada list.
2. Pilih satu sel pada list, pilih secara random tetangga sel yang belum dikunjungi, tambahkan tetangga tersebut pada list. Jika tidak ada tetangga yang bisa dikunjungi lagi hapus sel dari list.
3. Ulangi langkah 2 sampai list tersebut kosong [6].

BAB III

ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis sistem secara umum yang akan dibangun pada perangkat lunak ini. Selanjutnya dibahas mengenai perancangan permainan yang dibuat. Perancangan direpresentasikan dengan diagram UML (*Unified Modelling Language*) dan FSM (*Final State Machine*).

3.1. Analisis Sistem

Video game mempunyai berbagai macam *genre*, salah satunya adalah *roleplaying game*. *Roleplaying game* memiliki kelebihan tersendiri sehingga bila dimainkan tidak akan membuat bosan pemain, kelebihan tersebut adalah dengan membiarkan pemain mengatur karakter dalam pengaturan fiksi. Dalam pengaturan fiksi ini pemain biasanya akan diberikan *point* saat berhasil menaikkan level karakter, lalu *point* tersebut bisa digunakan untuk meningkatkan salah satu *attribute* dari karakter yang dimainkan oleh pemain. Misalkan saja jika dalam karakter yang dimainkan pemain terdapat *health*/darah, maka pemain bisa meningkatkan jumlah *health*/darah dengan menambahkan *point* yang telah diberikan kepada pemain. Sehingga pemain bisa mewujudkan karakter bagaimana yang ingin digunakan pada permainan, apakah karakter dengan darah yang banyak atau dengan *attack* yang besar.

Namun jika pemain mengulang kembali permainan dari awal pemain mungkin akan merasa bosan karena pemain akan kembali bermain pada lingkungan yang sama. Untuk menghindari ini maka akan dibangun *maze* yang akan berubah jika permainan diulang dari awal, diharapkan dengan lingkungan yang dinamis pemain tidak cepat merasa bosan karena permainan akan menjadi lebih menantang dan menarik.

Penulis membangun *video game* ini untuk *computer/desktop* dan menggunakan *game engine* Unity 3D dengan bahasa pemrograman C#. *Video game* ini mengambil

genre roleplaying game dan dibangun dengan *maze generator* yang bisa membangun *maze* yang variatif sehingga bisa membuat pemain betah untuk memainkan permainan ini.

3.2. Perancangan Permainan

3.2.1. Deskripsi Umum Perangkat Lunak

Video game yang akan dikembangkan pada tugas akhir ini adalah sebuah *video game* 3D dengan *genre roleplaying game* dengan lingkungan tempat bermain mengambil tema *maze*. Di dalam permainan ini pemain akan mengendalikan satu karakter yang memiliki beberapa *attribute* dan bisa memakai beberapa *item*. Untuk menyelesaikan permainan ini pemain harus bisa keluar dari *maze* di setiap level permainannya.

Karakter yang dikendalikan pemain akan memiliki beberapa *attribute* yaitu, *health* mempresentasikan jumlah darah pemain(jika darah lebih kecil atau sama dengan 0 maka pemain dianggap mati/kalah), *mana* untuk mempresentasikan jumlah *mana* maksimal yang dimiliki pemain(*mana* ini berguna untuk mengeluarkan *magic/skill*), *attack* untuk mempresentasikan jumlah *attack*(jumlah *damage* yang akan dikeluarkan pemain saat menyerang musuh), dan *defense* untuk mempresentasikan jumlah *defense* yang akan berguna untuk menahan serangan dari musuh.

Di dalam permainan terdapat 2 jenis *item*, yaitu *item* yang bisa dikonsumsi dan *item* yang tidak bisa dikonsumsi. *Item* yang bisa dikonsumsi seperti *health potion* yang apabila dikonsumsi akan menambah darah pemain, *mana potion* akan menambah *mana* bila digunakan dan *experient bonus* akan menambah jumlah *experient*, *experience* ini berguna untuk menaikkan level karakter. Selanjutnya *item* yang tidak bisa dikonsumsi terdiri dari senjata yang apabila dipakai dapat menaikkan *attack* dari karakter dan *item* yang bisa menaikkan *defense* yaitu *item* pelindung kepala, *item* pelindung kaki, *item* pelindung tangan dan *item* pelindung dada.

Di dalam permainan akan terdapat beberapa level yang harus dilewati agar menyelesaikan permainan ini. Disetiap levelnya akan terdapat *maze* dengan luas yang berbeda-beda. *Maze* ini akan dibuat dengan *maze* generator yang bisa membuat *maze* secara acak. Jika pemain memulai permainan baru, *maze* generator akan membuat *maze* secara acak dan menghasilkan *maze* yang berbeda dengan sebelumnya. Untuk keluar dari *maze* ini pemain harus mengumpulkan *key*/kunci yang terdapat pada peti yang penempatannya juga diacak pada permainan. Setelah mengumpulkan semua kunci pemain harus menuju pintu keluar yang penempatannya juga di *generate* secara acak di dalam *maze*.

3.2.2. Spesifikasi Kebutuhan Fungsional

Kebutuhan fungsional untuk *video game* ini hanya satu yaitu bisa dimainkan oleh pengguna.

3.2.3. Spesifikasi Kebutuhan Non-Fungsional

1. *FrameRate*

Framerate yang bagus merupakan kebutuhan standar untuk semua *video game*. Karena jika *framerate* terlalu kecil akan menyebabkan permainan melambat/*lag*, sehingga mengganggu kenyamanan permainan.

2. Grafis

Dikarenakan *video game* ini dibangun pada 3D dan *maze* yang dibangun terdiri dari banyak objek dibutuhkan grafis yang mencukupi agar permainan tidak melambat/*lag* yang bisa mengganggu kenyamanan memainkan *video game* ini.

3.2.4. Karakteristik Pengguna

Karakteristik pengguna tercantum pada Tabel 3.1. Hak akses yang akan diberikan kepada pemain hanya memainkan permainan, pemain tidak akan diberi hak akses untuk mengubah konten dari permainan. Di dalam permainan tidak tersedia bagaimana cara memainkan permainan ini, namun akan pemain dapat melihat tombola pa saja yang bisa dipakai dalam permainan ini.

Tabel 3.1. Karakteristik Pengguna

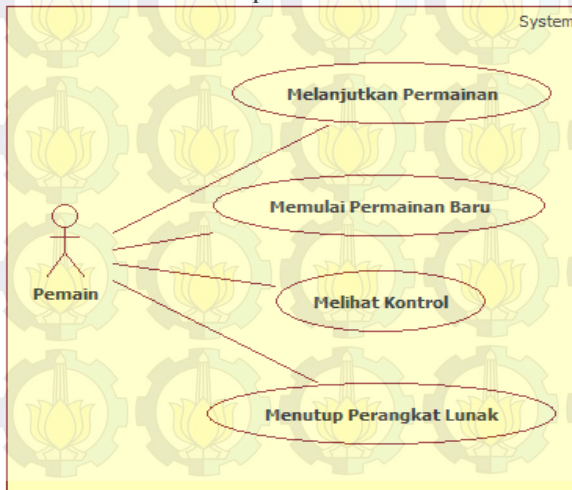
Nama Aktor	Tugas	Hak Akses Perangkat Lunak	Kemampuan yang Harus Dimiliki
Pemain	Memainkan permainan	Memainkan permainan	Mengerti bahasa Inggris dasar, pernah bermain <i>video game</i> dengan <i>genre roleplaying game</i> (tidak harus, tetapi sangat disarankan)

3.3. Perancangan Sistem

Perancangan sistem dibagi menjadi beberapa bagian, yaitu kasus penggunaan menu utama, kasus penggunaan dalam permainan, *control*, antarmuka dan *maze* generator.

3.3.1. Kasus Penggunaan Menu Utama

Bagian ini akan menjelaskan perancangan saat pengguna berada pada menu utama. Kasus penggunaan saat menu utama dicantumkan pada Gambar 1.



Gambar 1 Diagram kasus Penggunaan Menu Utama

Tabel 3.2. Kode Kasus Penggunaan Menu Utama

No	Kode Kasus Penggunaan	Nama Kasus Penggunaan	Keterangan
1	UC-001	Melanjutkan permainan	Untuk melanjutkan permainan dari data yang tersimpan terakhir
2	UC-002	Memulai permainan baru	Untuk memulai permainan dari awal
3	UC-003	Melihat control	Untuk melihat kontrol yang bisa dipakai dalam permainan
4	UC-004	Menutup Perangkat Lunak	Untuk menutup perangkat lunak

3.3.1.1. Spesifikasi Kasus Penggunaan Menu Utama

Berikut merupakan penjelasan kasus penggunaan menu utama. Terdapat 4 kasus penggunaan dalam menu utama, yaitu melanjutkan permainan, memulai permainan baru, melihat control dan menutup perangkat lunak.

Tabel 3.3. Melanjutkan Permainan

Nama	Melanjutkan permainan
Kode	UC-001
Deskripsi	Melanjutkan permainan dari data yang terakhir disimpan
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 1. Pemain menekan tombol Load Game 2. Sistem mengambil data yang tersimpan dan membangkitkan <i>scene</i> sesuai data yang tersimpan 3. Selesai

Tabel 3.4. Memulai Permainan Baru

Nama	Memulai permainan baru
Kode	UC-002
Deskripsi	Memilih menu permainan baru
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 1. Pemain menekan tombol New Game 2. Sistem menghapus data lama 3. Sistem menampilkan scene Creating New Character 4. Pemain mengisi persyaratan yaitu, mengisi nama karakter, memakai <i>point attribute</i> dan memilih <i>element</i> 5. Sistem memeriksa semua persyaratan sudah terisi. Sistem menampilkan tombol Lets Begin 6. Selesai <p>A.1. Pemain tidak mengisi persyaratan dengan lengkap</p>
Alur Alternatif	<p>A.1. Pemain tidak mengisi persyaratan dengan lengkap</p> <p>A.1.1. Kembali ke alur 3</p>

Tabel 3.5. Melihat Kontrol

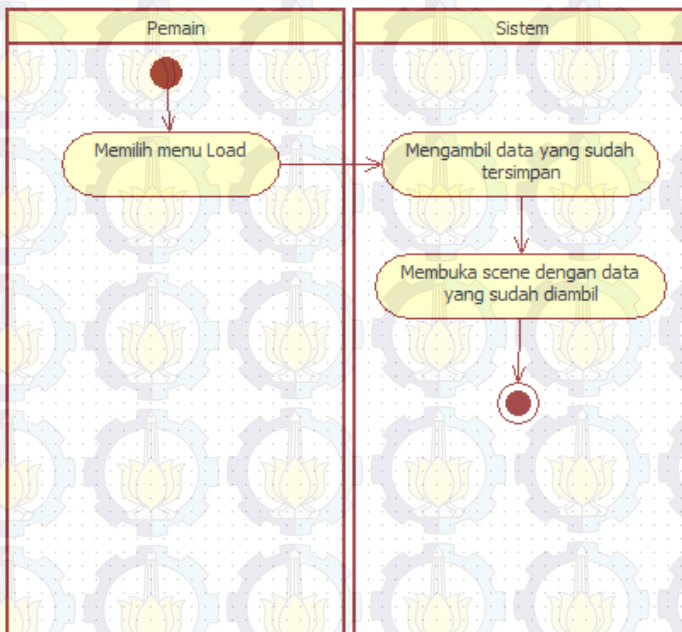
Nama	Melihat kontrol
Kode	UC-003
Deskripsi	Melihat tombol apa saja yang bisa dipakai pada permainan
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 4. Pemain menekan tombol Control 5. Sistem menampilkan scene Control 6. Selesai

Tabel 3.6. Menutup Perangkat Lunak

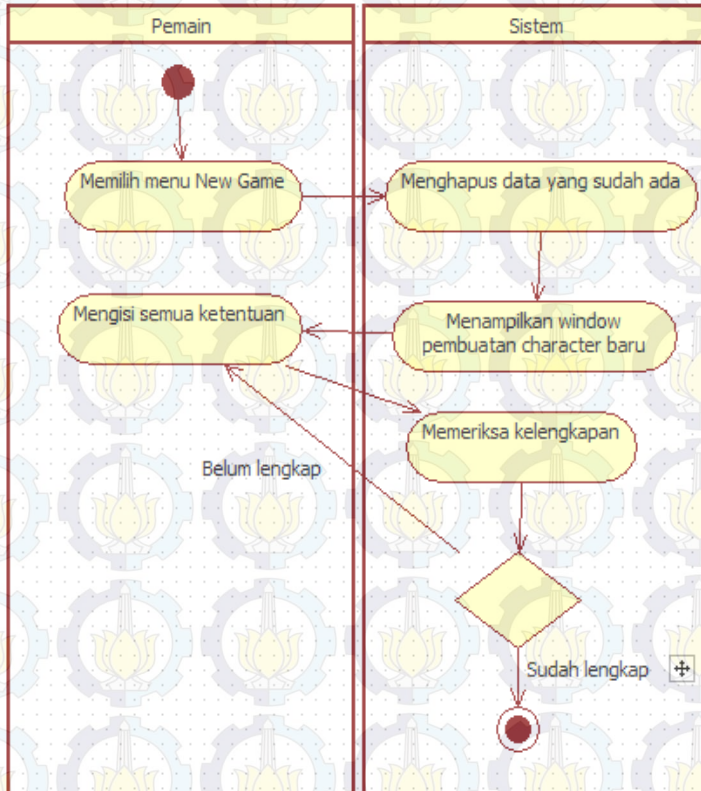
Nama	Menutup Perangkat lunak
Kode	UC-004
Deskripsi	Menutup perangkat lunak
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 1. Pemain menekan tombol Exit 2. Sistem menampilkan scene Control 3. Selsesai

3.3.1.2. Diagram Aktifitas Penggunaan Menu Utama

Berikut merupakan diagram aktifitas dari kasus penggunaan menu utama.

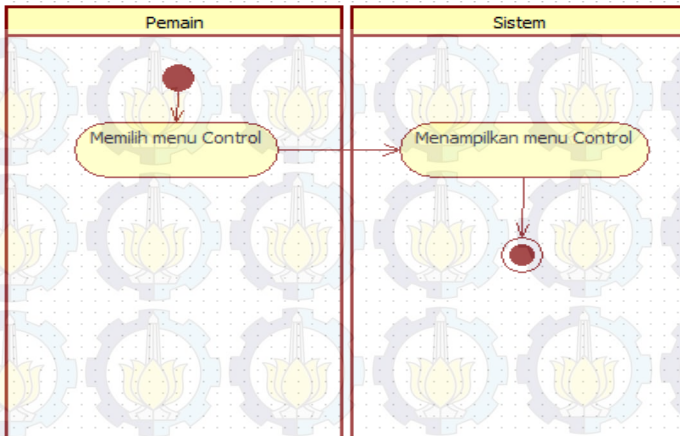
**Gambar 2. Diagram Aktifitas Melanjutkan Permainan**

Gambar 2 menunjukkan diagram aktifitas untuk melanjutkan permainan. Saat pemain memilih menu Load, sistem akan membaca data yang telah tersimpan lalu membuat scene berdasarkan data tersebut.



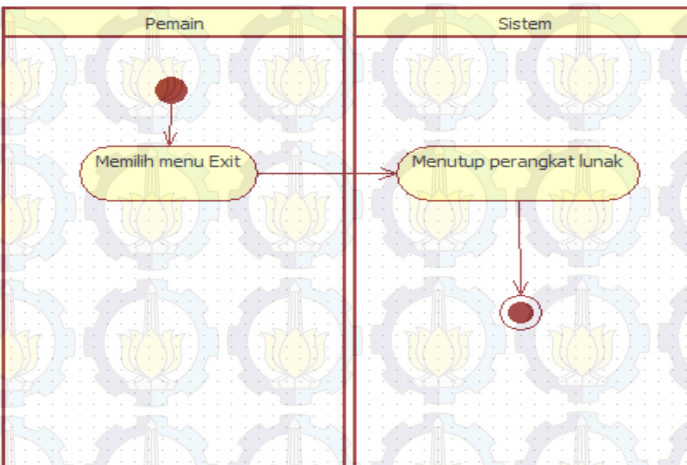
Gambar 3. Diagram Aktifitas Memulai Permainan Baru

Gambar 3 menunjukkan diagram aktifitas saat pemain memilih menu New Game. Sistem akan menghapus data yang sudah. Kelengkapan yang dimaksud adalah mengisi nama dan memakai *point attribute*. Pemain harus mengisi kelengkapan terbut, jika kelengkapan ini tidak terpenuhi pemain tidak akan bisa melanjutkan ke tahap selanjutnya.



Gambar 4. Diagram Aktivitas Melihat Kontrol

Gambar 4 diagram aktivitas saat pemain memilih menu Control.

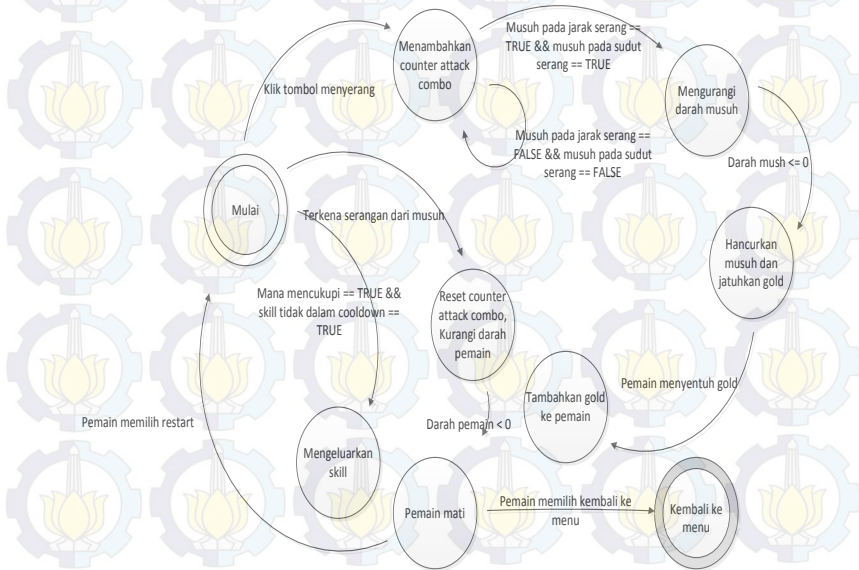


Gambar 5. Diagram Aktivitas Menutup Perangkat Lunak

Gambar 5 merupakan diagram aktivitas saat pemain ingin keluar dari permainan.

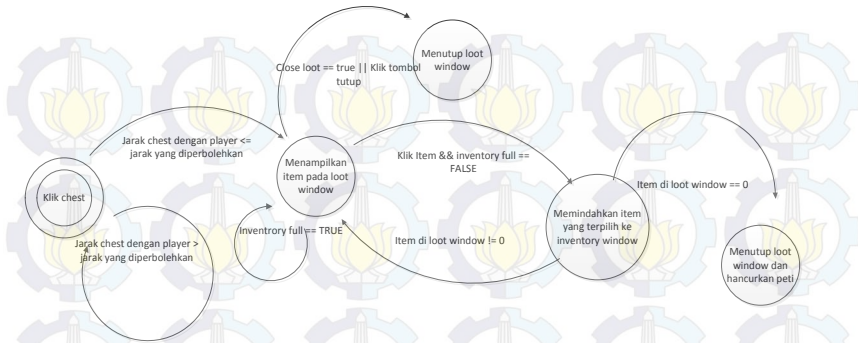
3.3.2. Kasus Penggunaan Saat Dalam Permainan

Dalam perancangan kasus penggunaan dalam permainan akan dijelaskan dengan FSM.



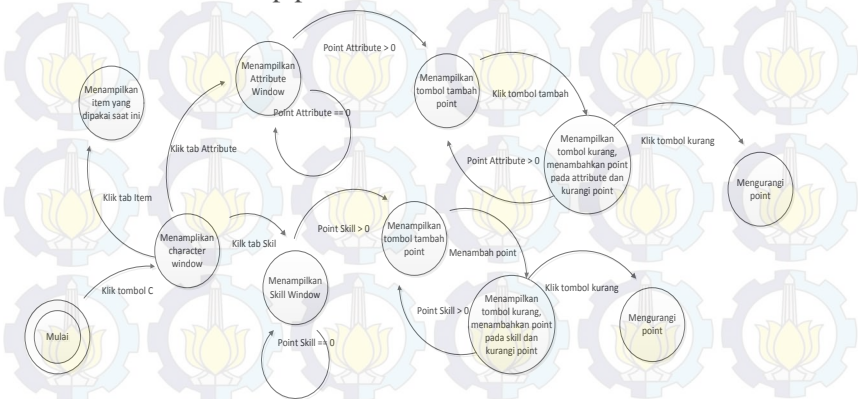
Gambar 6. FSM Pemain Menyerang

Gambar 6 menjelaskan saat pemain menekan tombol serang. Saat pemain menekan tombol serang maka *counter* untuk *attack combo* bertambah dan jika ada musuh dalam jangkauan serang maka kurangi darah musuh tersebut. Jika darah musuh kurang dari atau sama dengan 0 maka hancurkan musuh dan jatuhkan *gold* untuk pemain. *Gold* akan ditambahkan ke pemain jika pemain menyentuh *gold* tersebut. Namun jika saat menyerang pemain terkena serangan dari musuh maka *counter attack combo* direset menjadi 0. Pemain bisa mengeluarkan skill jika mana mencukupi dan *skill* tidak sedang *cooldown*.



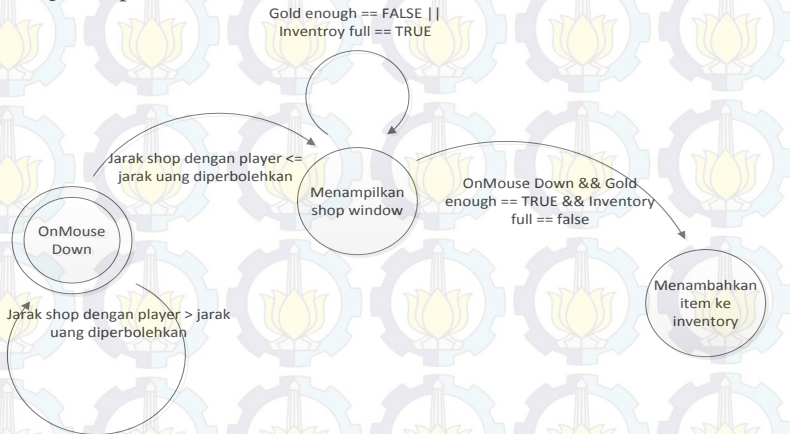
Gambar 7. FSM Membuka Peti

Pada Gambar 7 menjelaskan saat pemain membuka peti. Peti hanya bisa dibuka bila jarak pemain dengan peti tidak melebihi yang telah ditentukan. Saat pemain membuka peti, peti akan mengacak jumlah *item* yang akan diberikan. *Item* akan ditampilkan di *loot window*. Pemain bisa mengambilnya dengan klik *item* pada loot Window jika *inventory* pemain tidak penuh. Peti akan tertutup secara otomatis jika *item* pada 0 atau pemain jarak menjauhi peti. Pemain juga bisa menutup peti dengan menekan tombol tutup pada *loot window* nanti.



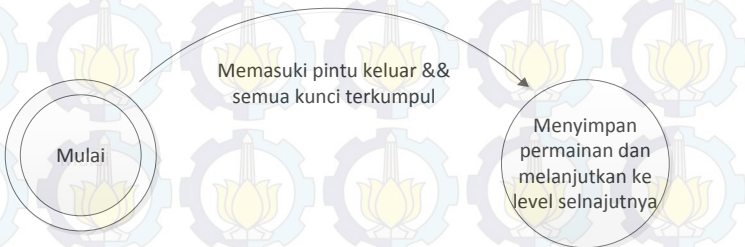
Gambar 8. FSM Character Window

FSM pada Gambar 8 menjelaskan alur saat pemain membuka character window. Pada character window terdapat 3 tab, yaitu tab *item*, *attribute* dan *skill*. Pada tab *item* terdapat *item* apa saja yang sedang dipakai oleh pemain. Lalu tab *attribute* digunakan untuk melihat dan memperbarui status *attribute* pemain. Yang terakhir tab *skill* untuk melihat dan memperbarui *skill point* pemain.



Gambar 9. FSM Berbelanja

Pada Gambar 9 menjelaskan saat pemain akan berbelanja. Pemain harus berada pada jarak tertentu untuk dapat membuka *window* berbelanja. Untuk berbelanja pemain harus menukarkan gold sesuai dengan harga barang yang ingin dibeli. Dan jika *inventory* pemain tidak penuh.



Gambar 10. FSM Keluar dari Maze

Gambar 10 menjelaskan saat pemain keluar dari *maze*. Pemain akan keluar dari maze dan melanjutkan permainan apabila pemain telah mengumpulkan semua kunci yang akan tersembunyi di salah satu peti dan melewati pintu keluar.

3.3.3. Kontrol

Di dalam permainan ini terdapat beberapa tombol yang bisa digunakan yaitu:

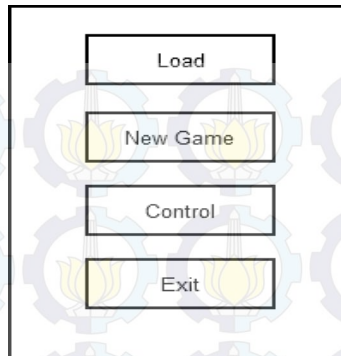
1. Tombol W untuk menggerakkan karakter maju.
2. Tombol S untuk menggerakkan karakter mundur/bergerak ke belakang.
3. Tombol A dan D untuk berputar.
4. Tombol C untuk memunculkan dan menyembunyikan karakter *window*.
5. Tombol I untuk memunculkan dan menyembunyikan *inventory window*.
6. Tombol P untuk memberhentikan sementara permainan.
7. Tombol 1 dan 2 untuk menggunakan *skill*.
8. Tombol kanan mouse untuk menyerang.
9. Tombol kiri mouse untuk membuka peti, mengambil barang, membeli *item* dan menggunakan *item*.
10. *Double click* tombol kiri mouse untuk memakai item.

3.3.4. Antarmuka

Dalam subbab ini diperlihatkan beberapa rancangan antarmuka dalam permainan.

3.3.4.1. Antarmuka Menu Utama

Pada Gambar 11 menunjukkan rancangan pada menu utama pada permainan. Terdapat 4 tombol yaitu *Load* untuk melanjutkan permainan dari data yang tersimpan terakhir. Untuk memulai permainan baru. *New Game* untuk memulai permainan baru. *Control* untuk melihat tombol apa saja yang bisa dipakai dalam permainan. *Exit* untuk keluar dari permainan.



Gambar 11. Perancangan Menu Utama

3.3.4.2. Antarmuka Menu Creation

A rectangular box representing the 'Menu Creation' interface. It contains the following elements:

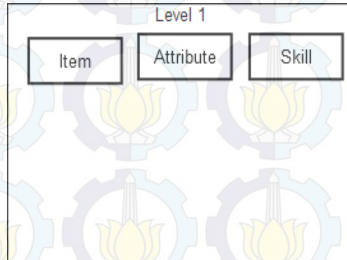
- Name :** A text input field.
- Point Left : 10** (Static text)
- Health : 10** (Static text) followed by a '-' button and a '+' button.
- Health : 10** (Static text) followed by a '-' button and a '+' button.
- Health : 10** (Static text) followed by a '-' button and a '+' button.
- Health : 10** (Static text) followed by a '-' button and a '+' button.
- Element :** Two buttons labeled 'Fire' and 'Earth' stacked vertically.
- Skill :** Two buttons labeled 'Skill1' and 'Skill 2' stacked vertically.
- Creating** (A button at the bottom right).

Gambar 12. Perancangan Menu Creation

Pada Gambar 12 merupakan rancangan Menu *Creation*. Menu ini akan muncul setelah pemain memilih menu *New Game* .

Pada menu ini pemain harus mengisi nama, menggunakan poin dan memilih element untuk melanjutkan permainan.

3.3.4.3. Antarmuka Character Window



Gambar 13. Perancangan Character Window

Gambar 13 menunjukkan Character Window, terdapat 3 tab yaitu tab Item untuk melihat item apa saja yang sedang dipakai oleh pemain. Tab Attribute untuk melihat *attribute* pemain saat ini, di dalam tab ini bisa menambahkan *point attribute* jika pemain mempunyai *point attribute*. Tab skill untuk melihat dan menambahkan *point skill* jika pemain mempunyai *point skill*.

3.3.4.4. Antarmuka Inventory Window



Gambar 14. Perancangan Inventory Window

Pada Gambar 14 menunjukkan Inventory Window untuk dapat menggunakan *item* pada *inventory window* pemain dapat menekan 2 kali pada item yang ingin dipakai.

3.3.4.5. Antarmuka Loot Window



Gambar 15. Perancangan Loot Window

Pada Gambar 15 menunjukkan Loot Window yang akan berisi *item-item* yang bisa diambil oleh pemain dengan menekan pada *item* yang ingin diambil. *Window* ini akan muncul jika pemain membuka peti.

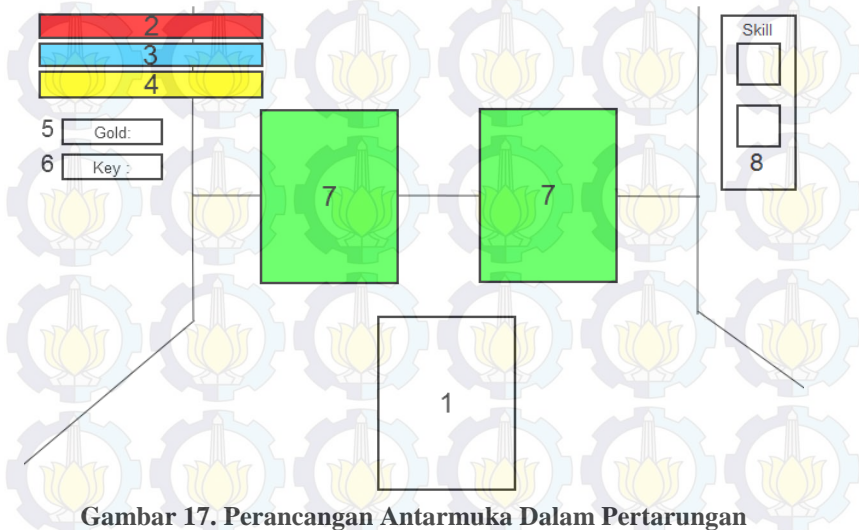
3.3.4.6. Antarmuka Shop Window



Gambar 16. Perancangan Antarmuka Shop Window

Gambar 16 menunjukkan Shop Window. *Window* ini akan muncul jika pemain menekan sebuah patung penjual yang akan ada pada permainan.

3.3.4.7. Antarmuka Dalam Pertarungan



Gambar 17. Perancangan Antarmuka Dalam Pertarungan

Keterangan pada Gambar 17 :

1. Posisi karakter pemain.
2. Bar darah pemain saat ini.
3. Bar mana pemain saat ini.
4. Bar exp/experient point pemain saat ini.
5. Menunjukkan *gold* pemain saat ini.
6. Menunjukkan *gold* pemain saat ini.
7. Posisi musuh.
8. Memberitakan *skill* apa saja yang dimiliki pemain saat ini.

Dalam permainan pemain akan mengambil *view third person*, yang artinya pemain akan bisa melihat karakternya sendiri.

3.3.5. Maze Generator

Dalam pembuatan *maze* tersebut menggunakan algoritma *Growing Tree*. Pertama akan dibangun sebuah kumpulan sel kubus seperti yang jumlahnya ditentukan diawal.

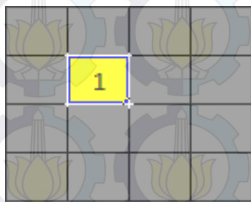


Gambar 18. Kumpulan Sel Kubus

1. Pilih secara acak satu sel, dan letakkan pada list.
2. Pilih satu sel pada list, pilih secara random tetangga sel yang belum dikunjungi, tambahkan tetangga tersebut pada list. Jika tidak ada tetangga yang bisa dikunjungi lagi hapus sel dari list.
3. Ulangi langkah 2 sampai list tersebut kosong.

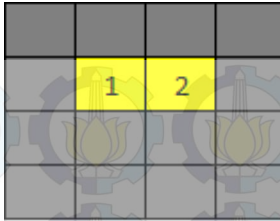
Contoh 1 pada sel kubus 4x4 jalannya algoritma *Growing Tree*:

1. Pilih satu sel dan letakkan pada list.



Gambar 19 Langkah 1 Algoritma Pada Sel Kubus 4x4

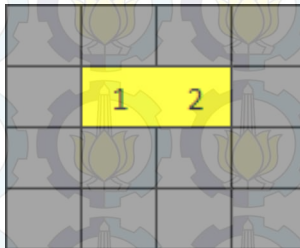
2. Pilih secara random tetangga dari sel 1,



	1	2	

Gambar 20 Langkah 2 Algoritma Pada Sel Kubus 4x4

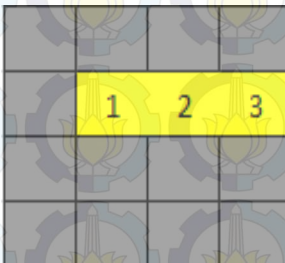
Tambahkan sel 2 pada list dan hilangkan dinding diantara sel ini.



	1	2	

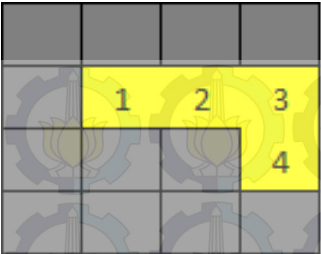
Gambar 21 Langkah 3 Algoritma Pada Sel Kubus 4x4

3. Ulangi langkah 2 sampai tidak ada tetangga yang bisa dikunjungi.



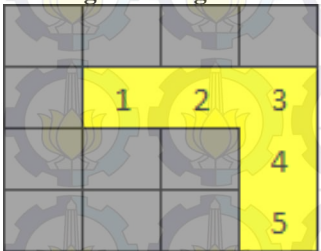
	1	2	3

Gambar 22 Langkah 4 Algoritma Pada Sel Kubus 4x4



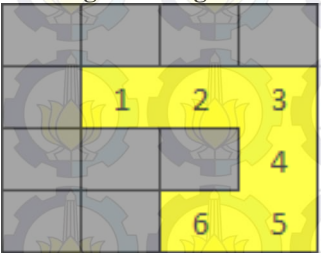
	1	2	3
			4

Gambar 23 Langkah 5 Algoritma Pada Sel Kubus 4x4



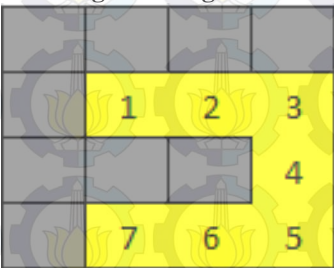
	1	2	3
			4
			5

Gambar 24 Langkah 6 Algoritma Pada Sel Kubus 4x4



	1	2	3
			4
		6	5

Gambar 25 Langkah 7 Algoritma Pada Sel Kubus 4x4



	1	2	3
			4
	7	6	5

Gambar 26 Langkah 8 Algoritma Pada Sel Kubus 4x4

	1	2	3
	8		4
	7	6	5

Gambar 27 Langkah 9 Algoritma Pada Sel Kubus 4x4

	1	2	3
9	8		4
	7	6	5

Gambar 28 Langkah 10 Algoritma Pada Sel Kubus 4x4

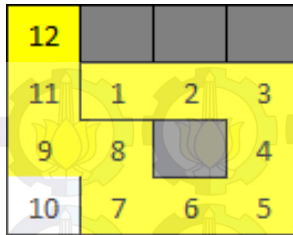
	1	2	3
9	8		4
10	7	6	5

Gambar 29 Langkah 11 Algoritma Pada Sel Kubus 4x4

4. Karena sudah tidak ada tetangga yang bisa dikunjungi maka buang sel 10 dari list. Lalu ambil sel 9 dan lakukan langkah 3 kembali.

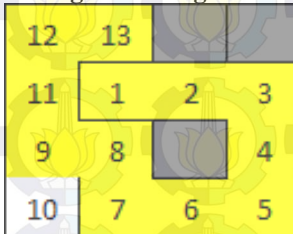
11	1	2	3
9	8		4
10	7	6	5

Gambar 30 Langkah 11 Algoritma Pada Sel Kubus 4x4



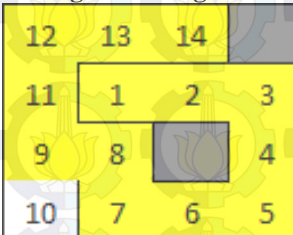
12			
11	1	2	3
9	8		4
10	7	6	5

Gambar 31 Langkah 12 Algoritma Pada Sel Kubus 4x4



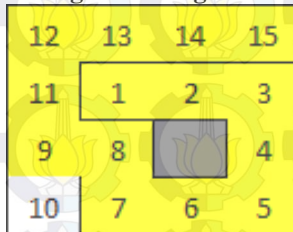
12	13		
11	1	2	3
9	8		4
10	7	6	5

Gambar 32 Langkah 13 Algoritma Pada Sel Kubus 4x4



12	13	14	
11	1	2	3
9	8		4
10	7	6	5

Gambar 33 Langkah 14 Algoritma Pada Sel Kubus 4x4



12	13	14	15
11	1	2	3
9	8		4
10	7	6	5

Gambar 34 Langkah 15 Algoritma Pada Sel Kubus 4x4

5. Karena sel 15 tidak ada tetangga lagi, maka sel 15 dihapus dari list. Sel dibuang dari list sampai sel 8, karena sel ini masih memiliki tetangga.

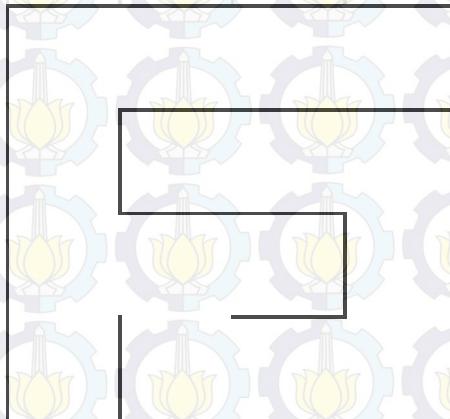
12	13	14	15
11	1	2	3
9	8		4
10	7	6	5

Gambar 35 Langkah 16 Algoritma Pada Sel Kubus 4x4

12	13	14	15
11	1	2	3
9	8	16	4
10	7	6	5

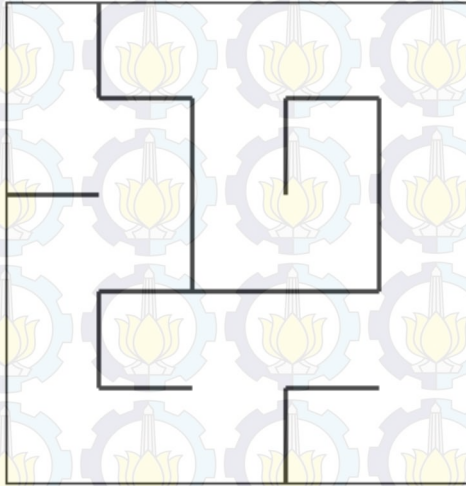
Gambar 36 Langkah 17 Algoritma Pada Sel Kubus 4x4

6. Karena sudah tidak ada tetangga yang belum dikunjungi iterasi dihentikan ditandai dengan list yang kosong.



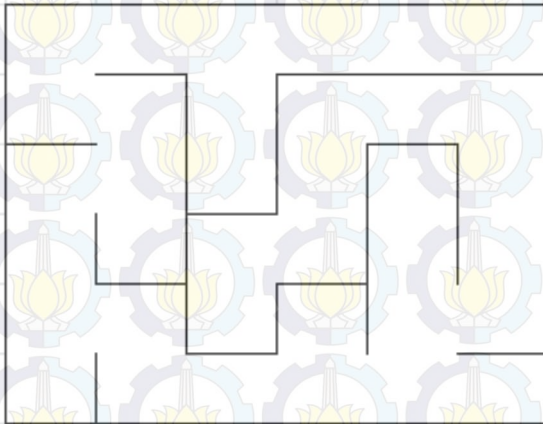
Gambar 37 Hasil Maze Dari Algoritma

Berikut contoh lain maze hasil dari algoritma *Growing Tree*:



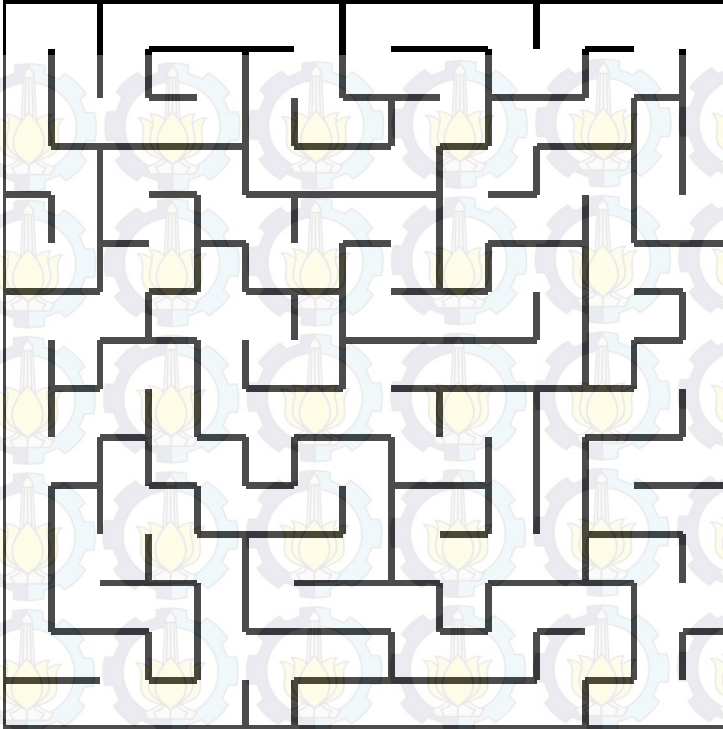
Gambar 38 Maze 5x5

Gambar 38 merupakan salah satu hasil maze dengan algoritma *Growing Tree* dengan ukuran 5x5.



Gambar 39 Maze 6x6

Gambar 39 merupakan salah satu hasil maze dengan algoritma *Growing Tree* dengan ukuran 6x6.



Gambar 40 Maze 15x15

Gambar 40 merupakan salah satu hasil maze dengan algoritma *Growing Tree* dengan ukuran 6x6.

Setelah maze selesai dibuat akan dilakukan random untuk penempatan monster dan peti/chest dengan jumlah yang telah ditentukan pada masing-masing level.

Untuk menjamin adanya pintu keluar, akan diletakkan sebuah objek sebagai pintu keluar secara acak.

3.3.6. Level dan Skenario

Berikut rancangan level dan skenario yang akan dibangun pada tugas akhir ini, ditunjukkan pada Tabel 3.7.

Tabel 3.7. Level dan Skenario

Level	Ukuran maze (Sel x Sel)
1	6 x 6
2	7 x 7
3	8 x 8
4	9 x 9
5	10 x 10
6	16 x 16
7	17 x 17
8	18 x 18
9	19 x 19
10	20 x 20
11	26 x 26
12	27 x 27
13	28 x 28
14	29 x 29
15	30 x 30
16	36 x 36
17	37 x 37
18	38 x 38
19	39 x 39
20	40 x 40

Pada Tabel 3.7 menunjukkan ukuran setiap *maze* setiap level. Pada level 1-5 pertambahan ukuran *maze* hanya satu ini dimaksudkan agar pemain bisa beradaptasi terlebih dahulu terhadap *maze*, namun untuk kelipatan 5 level ukuran *maze* ditambah 5 agar pemain lebih tertantang.

3.4. Design Asli dan Modifikasi *Growing Tree*

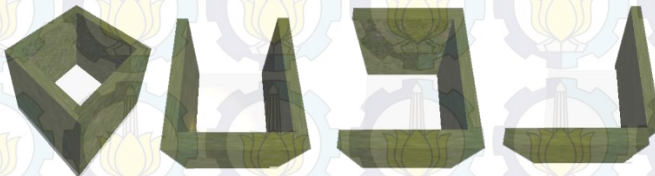
Dalam pembuatan kode untuk *Growing Tree* terdapat beberapa modifikasi baik untuk performance dalam jumlah objek ataupun menyesuaikan dengan kode di dalam permainan.

Pada design asli digunakan 1 jenis objek sebagai sel, objek tersebut ditunjukkan pada Gambar 41, potongan kode ditunjukkan pada Potongan Kode 8. Namun pada modifikasi objek yang digunakan sebagai sel ada 4 macam objek, objek ditunjukkan pada Gambar 42, sedangkan kode ditunjukkan pada Potongan Kode 2. Perubahan dilakukan karena pada design asli akan terjadi penumpukan 2 objek menjadi satu saat pembuatan kumpulan sel. Ini akan mengakibatkan jumlah objek lebih banyak dari seharusnya. Selanjutnya pada design asli pembuatan maze akan dimulai dengan suatu trigger ditunjukkan pada Potongan Kode 9, hal ini di modifikasi agar pembuatan maze dimulai tanpa memerlukan suatu trigger.

Modifikasi selanjutnya adalah pada design aslinya hasil dari maze tersebut tidak bisa disimpan, sehingga hal ini dimodifikasi agar hasil maze yang telah diciptakan bisa disimpan. Penyimpanan ini dilakukan agar maze yang dibuat tetap sama jika pemain memilih menu Load.



Gambar 41. Sel Pada Design Asli



Gambar 42. Semua Sel Pada Design Modifikasi

BAB IV IMPLEMENTASI

Bab ini membahas tentang implementasi dari perancangan sistem.

4.1. Lingkungan Implementasi

Berikut merupakan lingkungan yang digunakan saat implementasi.

Tabel 4.1. Lingkungan Implementasi

Perangkat Keras	Prosesor : Intel® CORE™ i5 CPU @ 2.53GHz Memori : 2 GB
Perangkat Lunak	Sistem Operasi : Microsoft Windows 7 64-bit Perangkat Pengembang : Unity

4.2. Implementasi Menu Utama

Pada implementasi Menu Utama terdapat 4 tombol yang bisa dipilih oleh pemain, yaitu Load, New Game, Control dan Exit. Berikut hasil implementasinya



Gambar 43. Tampilan Menu Utam

4.3. Implementasi Penggunaan Saat Permainan

Berikut merupakan *psedocode* dari beberapa FSM penggunaan saat permainan yang ada pada tahap perancangan.

```
if(Input.GetButtonDown("Attack") && !_attack &&
!_attacked){
    _forward = Forward.none;
    _attack = true;
    _animator.Play("Attack");
    if(_counterAttackAnimation == 3){
        _delayAttack = 2 * TIME_ATTACK;
    }
    _animator.SetFloat("CounterAnim",_counterAttack
Animation);
    _animator.SetBool("Attack",true);
    _speed = 0f;
    Combo();
}
```

Kode Sumber 1. Pemain menyerang

Pada Kode Sumber 1 menerapkan saat pemain menekan tombol serang/attack, pada fungsi *Combo* dilakukan penambahan *counter combo*. *Combo* ini akan berjalan apabila pemain bisa menyerang dalam kurun waktu yang telah ditentukan setelah serangan terakhir. Apabila pemain tidak bisa maka combo akan kembali menjadi 0.

```
If pemain menyerang
THEN
    If ada musuh pada jarak serang
    THEN
        Kurangi darah musuh tersebut
    ENDIF
ENDIF
```

Kode Sumber 2. Psedocode Mengurangi Darah Musuh

Kode Sumber 2 merupakan *psedocode* dari kode untuk mengurangi darah musuh. Darah musuh akan berkurang jika

pemain sedang menyerang dan musuh berada pada jarak serang pemain. Kode lengkap terlampir pada Potongan Kode 1.

```
private void DestroyThisObject() {
    if (_counterDestroy == 2f) {
        GetComponent<Animation>().Play("Die");
    }
    _counterDestroy -= Time.deltaTime;
    if (_counterDestroy <= 0) {
        GetComponent<Animation>().Stop("Die");
        Destroy(this.gameObject);
    }
}
```

Kode Sumber 3. Menghancurkan Objek Jika Mati

Kode Sumber 3 menunjukkan fungsi untuk menghancurkan objek saat mati.

```
Instantiate(gold, this.transform.position,
Quaternion.identity);
```

Kode Sumber 4. Membuat Objek Gold

Kode Sumber 4 digunakan untuk membuat objek *gold* saat musuh mati.

```
void CounterResetCombo() {
    if (_counterTimeCombo >= 0) {
        _counterTimeCombo -= Time.deltaTime;
    } else {
        _counterTimeCombo = 0;
    }
    if (_counterTimeCombo <= 0) {
        _counterAttackAnimation = 0;
    }
}
```

Kode Sumber 5. Reset Combo

Kode Sumber 5 fungsi untuk mereset *combo* saat pemain terkena serangan. *Combo* akan tereset apabila waktu yang telah ditentukan untuk pemain menyerang kembali telah habis.

```

public void DoDamage(float dmg){
    _attacked = true;
    Instantiate(bloodParticle, new
    Vector3(this.transform.position.x,
    this.transform.position.y + offsetSpawnBlood,
    this.transform.position.z),
    Quaternion.identity);
    _playerCharacter.HealthNow -= dmg;
    _animator.SetBool("Damaged",true);
    if(_playerCharacter.HealthNow <= 0){
        _animator.SetBool("Die",true);
        die = true;
        PlaySound(soundDie, false);
    }
}

```

Kode Sumber 6. Mengurangi Darah Pemain

Kode Sumber 6 digunakan untuk mengurangi darah pemain dan jika darah pemain kurang dari sama dengan 0 maka pemain mati.

```

private void DieWindow(int id){
    if(GUI.Button(new Rect(10, 30, 90, 30),
    "I'm Warrior")){
        Application.LoadLevel(PlayerPrefs.GetString("La
        st Level"));
    }
    if(GUI.Button(new Rect(110, 30, 90, 30),
    "Menu")){
        Application.LoadLevel("Menu");
    }
}

```

Kode Sumber 7. Window Saat Pemain Mati

Kode Sumber 7 digunakan untuk memunculkan *window* saat pemain mati. Jika pemain mati akan terdapat 2 pilihan, yaitu tombol *I'm warrior* dan *Menu*. Jika pemain memilih *I'm*

warrior maka sistem akan membangkitkan level terakhir pemain. Namun jika pemain memilih *Menu* maka pemain akan kembali ke menu utama.

```
void OnTriggerEnter(Collider c){
    if(c.tag == "Player"){
        GetComponent().Play();
        _showNotif = true;
        _tempAmount =
Random.Range(minAmount,maxAmount);
        _tempCounterHeight = _screenHeight -
20;

        Instantiate(goldParticle,
transform.position, Quaternion.identity);
c.gameObject.GetComponent<PlayerCharacter>().gold += _tempAmount;
this.gameObject.GetComponent<Collider>().enabled = false;

part[0].gameObject.GetComponent<SkinnedMeshRenderer>().enabled = false;
part[1].gameObject.GetComponent<MeshRenderer>().enabled = false;
        Destroy(this.gameObject,4f);
    }
}
```

Kode Sumber 8. Pemain Menyentuh *Gold*

Kode Sumber 8 saat pemain menyentuh *gold* tambahkan *gold* pemain sejumlah *gold* yang telah ditentukan dan hancurkan *gold* tersebut. *Gold* yang akan diterima oleh pemain tergantung dari nilai random yang keluar saat pertama kali *gold* dibuat. Selain itu, saat pemain mengambil *gold* tersebut akan dibuat *particle* berwarna kuning dan sebuah suara untuk menandakan bahwa pemain telah mengambil *gold* tersebut. Jadi pemain dapat mengetahui bahwa dirinya telah mengambil *gold* tersebut.

```

public void OnMouseUp() {
    if(!PlayerControl.pause){
        GameObject go =
        GameObject.FindGameObjectWithTag("Player");

        if(go == null){
            return;
        }

        if(Vector3.Distance(_myTransform.position,
        go.transform.position) > maxDistance &&
        !_inUse){

            return;
        }

        switch(state) {
            case Treasure.State.open:
                state = Treasure.State.inBetween;
                ForceClose();
                break;
            case Treasure.State.close:
                if(MyGUI.chest != null){
                    MyGUI.chest.ForceClose();
                }

                state = Treasure.State.inBetween;
                StartCoroutine("Open");
                break;
        }
    }
}

```

Kode Sumber 9. Membuka Peti

Kode Sumber 9 peti akan terbuka jika pemain memencet peti dan jarak kurang dari sama dengan jarak yang telah ditentukan. Setelah dibuka akan muncul Loot Window yang berisi item yang bisa diambil.


```

public void OnMouseUp() {
    if(!PlayerControl.pause){
        if(Vector3.Distance(transform.position,
        _player.transform.position) > maxDistance){
            return;
        }
        _animation.Play("Greeting");
        PlaySound(welcome, false);
        _playGreeting = true;
        openWindowShop = !openWindowShop;
        _openShop = true;
    }
}

```

Kode Sumber 10. *Shop*

Pada Kode Sumber 10 menunjukkan *shop* bisa dibuka saat jarak pemain dengan *shop* lebih kecil dari atau sama dengan jarak yang telah ditentukan.

```

void OnTriggerEnter(Collider c){
    if(c.tag == "Player"){
        Debug.Log("player");
        if(keyHaveNow == keyToExit){
            //save all
            _gsScripts.SaveCharacterData();
            //load next level
            int ne = _level.thisLevel + 1;
            PlayerPrefs.SetString("Last Level",
            "Level " + ne);
            string nextLevel = "Level " + ne;
            Application.LoadLevel(nextLevel);
        }
    }
}

```

Kode Sumber 11. *Exit*

Kode Sumber 11 menerapkan saat pemain memasuki pintu keluar jika kunci yang dikumpulkan sama dengan jumlah kunci yang telah ditentukan, simpan semua data terakhir dan lanjutkan ke level berikutnya.

4.4. Implementasi Kontrol

```

if(Input.GetButton("Horizontal") &&
!_attacked && !_attack ){
    if(Input.GetAxis("Horizontal") > 0){
        _turn = Turn.right;
    }else{
        _turn = Turn.left;
    }
}
if(Input.GetButton("Vertical") &&
!_attacked && !_attack){
}
if(Input.GetButtonDown("Attack") &&
!_attack && !_attacked){
}
if(Input.GetButtonUp("Toggle Inventory")){
    Messenger.Broadcast("ToggleInventory");
}
if(Input.GetButtonUp("Toggle Character
Window")){
    Messenger.Broadcast("ToggleCharacterWindow"
);
}
if(Input.GetButtonUp("Skill1") &&
_playerCharacter.allSkill[0].Available){
}
if(Input.GetButtonUp("Skill2") &&
_playerCharacter.allSkill[1].Available){
}
if(Input.GetButtonDown("Pause")){
    if(pause){
    }
    pause = !pause;
}

```

Kode Sumber 12. Kode Kontrol

Kode Sumber 12 menunjukkan untuk membuat kontrol pada permainan. Beberapa tombol terdapat syarat untuk bisa ditekan seperti *skill* terdapat syarat yaitu *skill* tersebut tidak dalam keadaan *cooldown* dan mana yang dimiliki pemain melebihi atau

sama dengan mana yang dibutuhkan untuk menggunakan *skill* tersebut.

4.5. Implementasi Antarmuka

4.5.1. Antarmuka Menu Character Creation



Gambar 44. Tampilan Character Creation

Gambar 44 merupakan hasil dari penerapan rancangan antarmuka menu Character Creation. Nama pemain bisa diisi dengan mengisi kolom nama yang terletak sebelah kiri label nama. Untuk menggunakan *point* dapat menekan tanda tambah disamping *attribute* yang ingin ditambahkan *point*.

4.5.2. Antarmuka Character Window

Pada Gambar 45 menunjukkan Character Window Tab Equipment. Terdapat 6 slot yang bisa di isi. 1 slot untuk weapon, 1 slot untuk pelindung kepala, 1 slot untuk pelindung dada, 2 slot untuk pelindung tangan dan 2 slot untuk pelindung kaki.



Gambar 45. Tampilan Character Window Tab Equipment



Gambar 46. Tampilan Tab Attributes Pada Character Window

Pada Gambar 46 menunjukkan 4 *attribute* yang dimiliki pemain, yaitu *health*, *mana*, *defense*, *attack*. Jika *point attribute* pemain sama dengan nol maka tombol pada samping *attribute* akan disembunyikan sehingga pemain tidak bisa menambahkan *point* pada *attribute* tersebut.



Gambar 47. Tampilan Point Attribute Jika Tidak Nol

Jika Point Attribute tidak nol maka akan muncul tombol tambah disamping *attribute*, seperti yang ditunjukkan pada Gambar 47. Pemain dapat menambahkan *point* ke *attribute* mana pun yang diinginkan oleh pemain. Dengan menambahkan *point* tersebut ke salah satu *attribute*, maka *attribute* tersebut akan memiliki nilai yang lebih besar dari sebelumnya.



Gambar 48. Tampilan Tab Skills Pada Character Window

Selanjutnya pada Gambar 48 menunjukkan tab Skills jika Point Skill sama dengan nol. Pemain tidak bisa menambahkan *point*.



Gambar 49. Tampilan Jika Point Skill Tidak Nol

Sedangkan Gambar 49 menunjukkan tab Skills apabila Point Skills sama dengan 1. Pemain dapat menambahkan *point* tersebut ke *skill* mana pun yang diinginkan. Dengan menambahkan *point* tersebut *skill* pemain tersebut akan memiliki efek/serangan yang lebih besar namun memerlukan mana yang lebih banyak juga.

4.5.3. Antarmuka Inventory Window



Gambar 50. Tampilan Inventory Window

Gambar 50 menunjukkan *inventory* pemain. Untuk saat ini *inventory* maksimal adalah 12 x 6. Jika *inventory* pemain penuh maka pemain dapat menghancurkan *item* yang tidak dipakai oleh pemain dengan memilih *item* tersebut lalu menekan tombol *destroy* pada pojok kanan atas *window*.

4.5.4. Antarmuka Loot Window



Gambar 51. Tampilan Loot Window

Gambar 51 merupakan Loot Window yang akan muncul apabila pemain membuka peti. Pemain dapat mengambil *item* dengan menekan *item* yang diinginkan. *Item* akan secara otomatis berpindah ke *inventory* jika *inventory* tidak penuh.

4.5.5. Antarmuka Shop Window



Gambar 52. Tampilan Shop Window

Pada Gambar 52 menunjukkan Shop Window yang mempunyai 3 tab, yaitu tab Consumable, Common, Uncommon dan Rare. Jika pemain ingin membeli maka pemain dapat menekan dua kali pada *item* yang diinginkan.

4.5.6. Antarmuka Dalam Pertarungan



Gambar 53. Tampilan Dalam Pertarungan

Gambar 53 menunjukkan hasil implementasi saat pertarungan, terdapat letak pemain, musuh, darah pemain, mana pemain, *exp* pemain saat ini, *gold* pemain saat ini, kunci yang perlu dikumpulkan pemain dan skill window yang terletak pada kanan atas layar.

4.6. Implementasi *Maze* Generator

```
public void CreateTheMaze () {
    CreateAllCell ();
    SetWhereBegin (0, 0);

    while (visitedCellList.Count != 0) {
        FindNextCell ();
    }
    PlayerPrefs.SetString ("Maze Level" +
    level, _nextMaze);
}
```

Kode Sumber 13. Implementasi *Maze* Generator

Kode Sumber 13 menunjukkan fungsi utama dalam pembuatan maze. Fungsi ini diawali dengan pembuatan kumpulan sel dengan jumlah yang telah ditentukan sebelumnya, tugas ini dijalankan oleh fungsi `CreateAllCell` (kode ditunjukkan pada Potongan Kode 2 di lampiran potongan kode). Selanjutnya ditentukan dimana pembuatan maze dimulai dengan fungsi `SetWhereBegin` (kode ditunjukkan pada Potongan Kode 4 di lampiran potongan kode). Lalu dijalankan fungsi `FindNextCell` (ditunjukkan pada Potongan Kode 3 di lampiran potongan kode) sampai list pada `visitedCellList` habis. Terakhir simpan data untuk membuat *maze* yang sama.

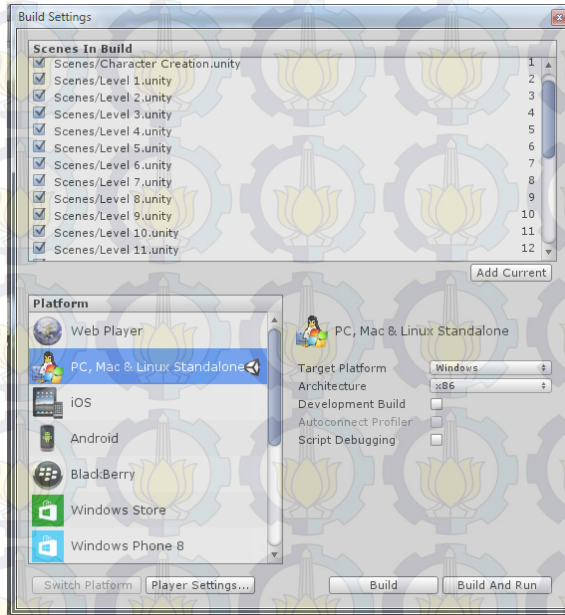
```
private void SpawnExit() {
    bool done = false;
    do {
        for (int i = 0; i <
            _allPlaceAvailable.Count; i++) {
            int rn = Random.Range(0, 3);
            if (rn == 1) {
                Instantiate(exitDoor, new
                Vector3((_allPlaceAvailable[i].x *
                _allcel.buffer), 0f, (_allPlaceAvailable[i].z *
                _allcel.buffer)), Quaternion.identity);
                done = true;
            }
        }
    } while (!done);
}
```

Kode Sumber 14. Membuat Pintu Keluar

Pintu keluar dibangun secara acak di dalam sel pada maze yang telah dihubungkan dengan minimal satu sel tetangganya, karena hal tersebut pemain dapat dipastikan bisa mencapai pintu keluar. Pada penempatan pintu keluar sudah dapat dipastikan saat meletakkan pintu keluar tidak akan sama dengan penempatan peti. Sehingga untuk sebelum mencapai pintu keluar pemain

harus mencari peti dengan kunci yang tersembunyi di dalamnya. Jika belum bisa menemukan kunci tersebut pemain tidak akan bisa keluar dari maze dan melanjutkan ke maze berikutnya dengan ukuran yang lebih besar.

4.7. Implementasi Level dan Skenario



Gambar 54. Screenshoot dari Unity Untuk Level 1-11

Gambar 54 merupakan pengaturan untuk level 1 sampai level 11 di dalam unity. Pada setiap *scene* yang dimasukkan dalam pengaturan level telah diatur semua ukuran *maze* sesuai dengan skenario yang telah dibuat. Pada *level 1-5* penambahan ukuran sisi *maze* hanya satu. Ini dimaksudkan agar pemain bisa beradaptasi dengan permainan. Level 6 memiliki perbedaan 5 pada ukuran sisi dengan level 5. Begitu juga dengan level kelipatan 5 berikutnya akan memiliki perbedaan ukuran sisi sebanyak 5.

Gambar 55 implementasi level 12 sampai dengan level 20 pada unity.



Gambar 55. Implementasi Level 12-20

4.8. Pembuatan Karakter

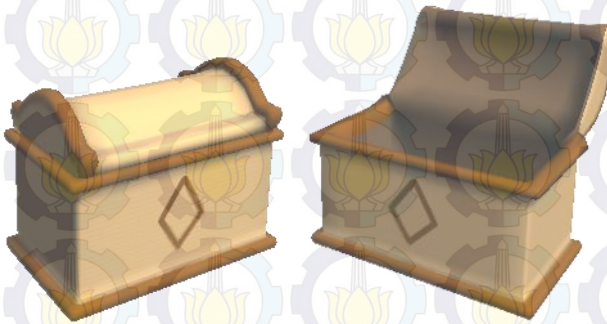
Pembuatan semua model 3D pada *video game* ini menggunakan perangkat lunak Blender. Sedangkan untuk pengerjaan texturnya menggunakan GIMP. Model terbagi menjadi 2 kelompok. Kelompok pertama model 3D yang terdapat animasinya dan model 3D tanpa animasi.

Model 3D yang menggunakan animasi antara lain model untuk karakter utama, monster, dan peti. Sedangkan model 3D yang tidak terdapat animasinya adalah model untuk senjata, gold, pelindung dada, tangan, kaki dan kepala. Untuk item yang bisa di konsumsi tidak dibuat model 3Dnya dikarenakan item tersebut hanya akan digunakan texture 2D saja pada Inventory Window atau Loot Window.



Gambar 56. Karakter Utama

Gambar 56 merupakan karakter utama yang akan dimainkan oleh pemain pada permainan. Beberapa animation yang ada pada karakter ini antara lain berjalan, menyerang, dan menggunakan skill.



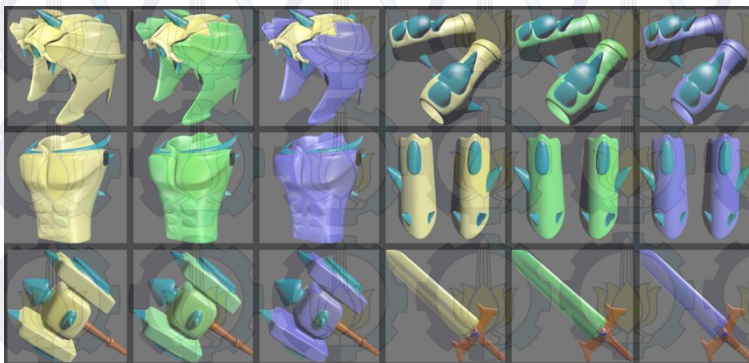
Gambar 57. Peti

Gambar 57 Merupakan model untuk peti yang bisa dibuka oleh pemain. Model ini memiliki animation membuka dan menutup peti. Jika peti ini dibuka maka akan muncul Loot Window, di dalam Loot Window tersebut terdapat item yang bisa diambil.



Gambar 58. Monster

Gambar 58 merupakan 3 model 3D untuk monster yang digunakan di dalam permainan. Animasi yang terdapat pada karakter adalah berjalan dan menyerang.



Gambar 59. All Item

Gambar 59 merupakan semua item yang bisa digunakan di dalam permainan. Terdapat 3 tingkatan item yaitu *common*, *uncommon* dan *rare* pada setiap jenis itemnya.

BAB V

PENGUJIAN DAN EVALUASI

Bab ini akan dijelaskan rangkaian uji coba dan evaluasi terhadap tugas akhir ini. Proses pengujian dilakukan menggunakan metode *blackbox* berdasarkan skenario yang telah ditentukan.

5.1. Lingkungan Uji Coba

Lingkungan pengujian sistem pada pengerjaan tugas akhir ini dilakukan pada lingkungan dan alat kaku sebagai berikut:

Tabel 5.1. Lingkungan Uji Coba

Perangkat Keras	Prosesor : Intel® CORE™ i5 CPU @ 2.53GHz Memori : 2 GB
Perangkat Lunak	Sistem Operasi : Microsoft Windows 7 64-bit Perangkat Pengembang : Unity

Di dalam dokumentasi Unity disebutkan OS yang bisa menjalankan hasil build dari Unity adalah Windows XP+, Mac OS X 10.7+, Ubuntu 12.04+, dan SteamOS+.

5.2. Skenario Uji Coba

Tabel 5.2 menunjukkan beberapa skenario yang akan diuji cobakan terhadap perangkat lunak yang telah dibuat. Terdapat 12 skenario yang akan diuji coba. Dua belas skenario dipilih karena dianggap bisa mempresentasikan fungsional perangkat lunak. Pengujian dilakukan dengan menjalankan perangkat lunak lalu mencoba menjalankan perangkat lunak sesuai dengan skenario yang telah dibuat.

Untuk pengujian pembuatan *maze* dibuat *scene* sendiri untuk pengujian dengan kamera yang ditaruh diatas *maze*, dikarenakan jika pengujian pada *scene* level menggunakan

kamera yang mengikuti karakter maka hasil dari maze tidak akan terlihat.

Tabel 5.2. Tabel Skenario Uji Coba

No	Nama	Keterangan
1	Menggunakan <i>Combo</i>	Pemain menyerang musuh sehingga <i>combo</i> keluar
2	Mengurangi Darah Musuh	Musuh menerima serangan pemain sehingga darah musuh berkurang dan menjatuhkan gold apabila darah musuh kurang dari atau sama dengan 0.
3	Mendapatkan <i>Gold</i>	Pemain melewati <i>gold</i> sehingga jumlah <i>gold</i> pemain bertambah
4	Pemain Mati	Pemain menerima serangan hingga darah pemain kurang dari 0 dan memunculkan <i>window</i> pemain mati
5	<i>Pause</i>	Pemain menekan tombol P untuk pause pada permainan
6	Memunculkan Character Window	Pemain menekan tombol C untuk memunculkan Character Window
7	Memunculkan Inventory Window	Pemain menekan tombol I untuk memunculkan Inventory Window
8	Membuka Peti	Pemain membuka peti
9	Mengambil Item	Pemain mengambil <i>item</i> dari Loot Window
10	Membeli Item	Pemain membuka <i>shop</i> lalu membeli <i>item</i> yang diinginkan
11	Pembuatan Maze	Mencoba membangun beberapa <i>maze</i> dalam beberapa kali coba untuk melihat apakah <i>maze</i> yang dibangun sudah berbeda
12	Peletakan <i>Exit</i>	Menunjukkan bahwa tempat <i>exit</i> bisa dicapai oleh pemain

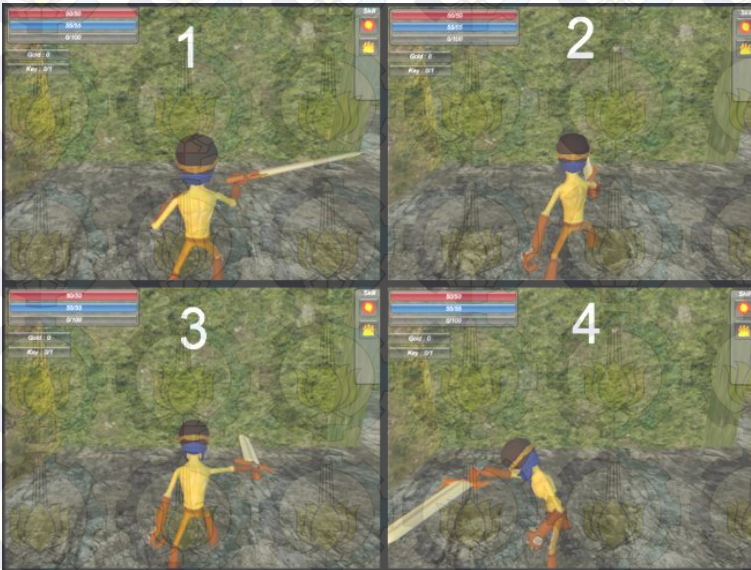
5.3. Hasil Uji Coba

Kondisi awal dari semua uji coba adalah pemain telah membuat karakter dan masuk dalam *maze*.

5.3.1. Menggunakan *Combo*

Pemain memiliki 4 serangan berbeda sebagai *combo*. Setiap kali pemain menyerang, *counter* untuk *combo* ini bertambah. Jika pemain terkena serangan atau *counter combo* sudah mencapai 4 maka *counter combo* menjadi 0. Berikut 4 serangan *combo* pada Gambar 60.

Pada serangan no 1 merupakan gerakan menebas dari kanan ke depan. Lalu pada serangan no 2 merupakan gerakan menusuk ke depan. Serangan no 3 seperti gerakan pada serangan no 1 namun gerakan no 3 menebas dari kir ke depan. Terakhir serangan no 4 merupakan gerakan menebas juga namun jangkuaangan serangan ini lebih lebar dari serangan no 1 dan 3.



Gambar 60. Semua Serangan *Combo*

5.3.2. Mengurangi Darah Musuh

Darah musuh akan berkurang jika terkena serangan biasa dari pemain atau skill dari pemain. Darah musuh berbentuk kubus berwarna yang berada di atas musuh. Jika kubus habis berarti darah musuh telah 0 ditunjukkan pada Gambar 61.

Jumlah darah yang berkurang merupakan jumlah penjumlahan *attack* pada karakter ditambah dengan jumlah *attack* yang terdapat pada senjata. Setiap senjata memiliki nilai *attack* minimal dan nilai *attack* maksimal. Setiap menyerang *attack* yang akan dikeluarkan senjata akan diacak antara nilai *attack* minimal dan nilai *attack* maksimal.



Gambar 61. Darah Musuh Berkurang

5.3.3. Mendapatkan Gold

Setelah membunuh musuh, maka akan terdapat gold yang dijatuhkan oleh musuh. Jumlah gold yang dijatuhkan diacak diantara dua nilai yang telah ditentukan.

Pada Gambar 62 jumlah awal gold pemain 0. Setelah menyentuh objek gold yang terdapat di atas tanah maka gold pemain bertambah 26 ditunjukkan pada Gambar 63.



Gambar 62. Gold Pemain 0



Gambar 63. Gold Pemain Bertambah

5.3.4. Pemain Mati

Saat darah pemain 0, maka akan muncul *die window*. Di dalam *window* ini terdapat pilihan untuk *respawn* kembali pada *maze* tersebut atau kembali ke menu utama. *Window* tersebut ditunjukkan pada Gambar 64. Di dalam *window* terdapat 2 tombol yaitu tombol I'm warrior untuk mengulang permainan pada level terakhir dan tombol Menu untuk kembali ke menu utama.



Gambar 64. Pemain Mati

5.3.5. Pause

Gambar 65 menunjukkan jika pemain menekan tombol P. Setelah menekan tombol *pause* ini, maka permainan akan berhenti untuk sementara. Di dalam *window pause* ini terdapat tombol Back to Main Menu untuk kembali ke menu utama permainan. Selanjutnya tombol Kontrol untuk melihat tombol apa saja yang bisa dipakai selama permainan. Untu *unpause* permainan pemain dapat menekan tombol P kembali.



Gambar 65. Pause

5.3.6. Memunculkan Character Window

Gambar 66 merupakan tampilan ketika pemain menekan tombol C. Terdapat 3 tab yang bisa dipilih Equipment untuk melihat item apa saja yang dipakai oleh pemain. Tab Attribute digunakan untuk melihat *attribute* dan menambahkan *point* jika memiliki Point Attribute. Terakhir tab Skills untuk melihat *skill* dan menambahkan *point* jika mempunyai *point*. Window ini bisa digeser sesuai keinginan pemain.



Gambar 66. Character Window

5.3.7. Memunculkan Inventory Window

Gambar 67 merupakan tampilan ketika pemain menekan tombol I. Inventory Window ini bisa digeser dan ditempatkan sesuai keinginan pemain. Untuk menutupnya bisa menekan tombol I kembali atau menekan tombol silang pada pojok kiri atas *window*.



Gambar 67. Inventory Window

5.3.8. Membuka Peti

Gambar 68 menunjukkan sebelum dan sesudah peti dibuka. Setelah membuka peti akan terbual Loot Window. Pada *window* ini akan terdapat item yang bisa diambil oleh pemain. Loot Window ini akan secara otomatis tertutup jika pemain berjalan menjauhi peti atau *item* yang terdapat pada *window* ini telah diambil semua. Selain itu pemain juga bisa menutup *window* ini secara manual dengan menekan tombol silang pada pojok kiri atas Loot Window. *Window* ini tidak bisa digeser seperti 2 *window* sebelumnya.



Gambar 68. Membuka Peti

5.3.9. Mengambil *Item*

Gambar 69 menunjukkan setelah membuka peti maka Loot Window akan muncul. Pemain bisa memilih mengambil mengambil *item* dengan menekan *item* yang diinginkan. Jika pemain mengambil semua *item* yang terdapat pada Loot Window maka *window* tersebut akan tertutup secara otomatis dan peti akan dihancurkan.

Gambar 69. Mengambil *Item* dari Loot Window

5.3.10. Membeli *Item*

Sebelum dapat membeli *item* pemain harus menekan penjual dan berada pada jarak yang cukup dekat untuk membuka Shop Window. Setelah membuka Shop Window pemain dapat membeli *item* yang diinginkan dengan *double left click* pada *item* tersebut ditunjukkan pada Gambar 70.



Gambar 70. Membeli Item

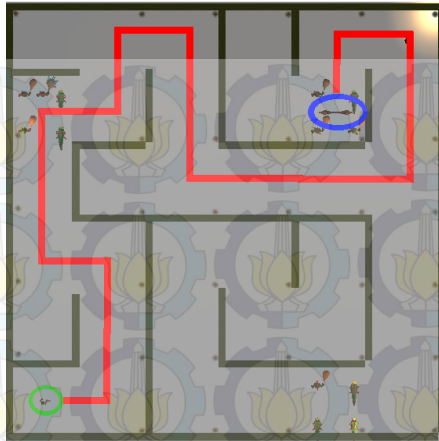
5.3.11. Pembuatan *Maze*

Berikut merupakan *maze* yang dihasilkan dari *maze* generator pada perangkat lunak ini. Setiap ukuran *maze* dicoba sebanyak 3 kali. Dan dihasilkan 3 *maze* dengan ukuran yang sama namun berbeda. Lingkaran hijau merupakan pemain dan lingkaran biru pintu keluar dari *maze* untuk melanjutkan ke *maze* berikutnya. Garis merah merupakan jalan dari pemain menuju pintu keluar.

Gambar 71 merupakan hasil pertama dari *maze* generator dengan ukuran 6 x 6.

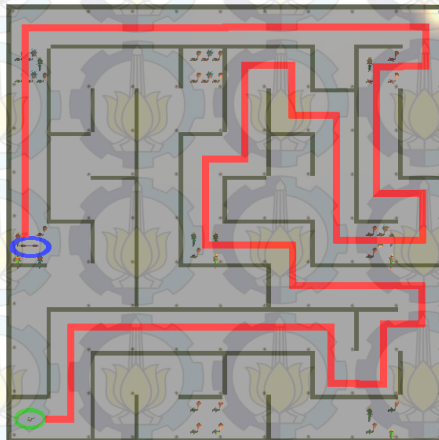
Gambar 72 merupakan hasil kedua dari *maze* generator dengan ukuran 6 x 6.

Gambar 72 merupakan hasil kedua dari *maze* generator dengan ukuran 6 x 6.



Gambar 73. Hasil 3 Maze 6x6

Gambar 73 merupakan hasil ketiga dari *maze* generator dengan ukuran 6 x 6.



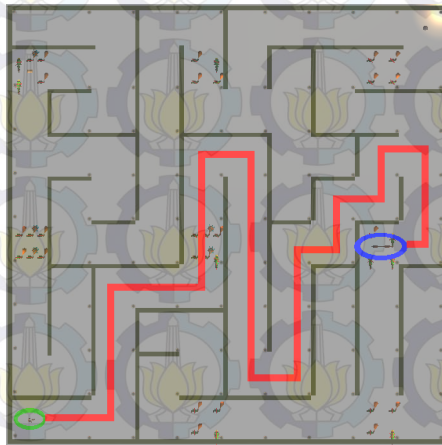
Gambar 74. Hasil 1 Maze 10x10

Gambar 74 merupakan hasil pertama dari *maze* generator dengan ukuran 10 x 10.



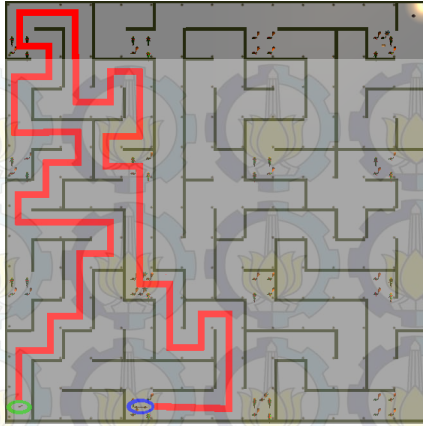
Gambar 75. Hasil 2 Maze 10x10

Gambar 75 merupakan hasil kedua dari *maze* generator dengan ukuran 10 x 10.



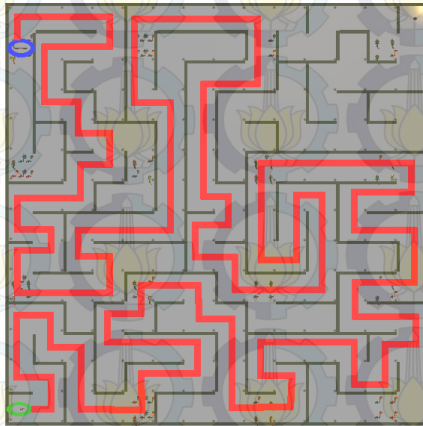
Gambar 76. Hasil 3 Maze 10x10

Gambar 76 merupakan hasil ketiga dari *maze* generator dengan ukuran 10 x 10.



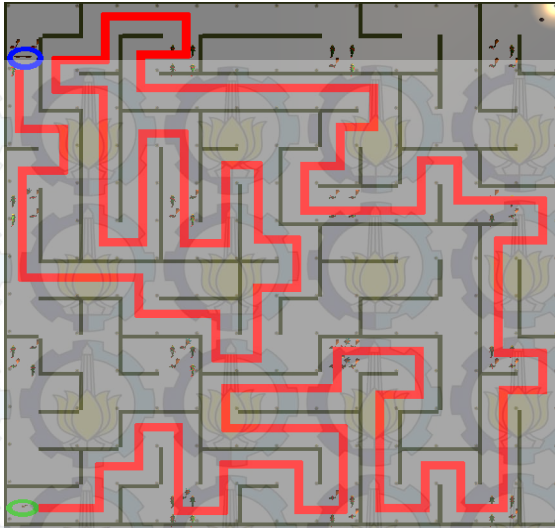
Gambar 77. Hasil 1 Maze 14x14

Gambar 77 merupakan hasil pertama dari *maze* generator dengan ukuran 14 x 14.



Gambar 78. Hasil 2 Maze 14x14

Gambar 78 merupakan hasil kedua dari *maze* generator dengan ukuran 14 x 14.



Gambar 79. Hasil 3 Maze 14x14

Gambar 79 merupakan hasil ketiga dari *maze* generator dengan ukuran 14 x 14.

5.4. Evaluasi

Dari uji coba yang telah dilakukan dapat disimpulkan bahwa semua skenario telah berjalan seperti yang diinginkan. Pada pembuatan *maze* dapat dilihat bahwa *maze* yang dibuat berbeda dan pintu keluar pasti bisa dicapai oleh pemain.

5.5. Kuisisioner

Dalam pengujian permainan ini diadakan suatu kuisisioner dengan range nilai 1-4, dengan ketentuan semakin besar nilai semakin baik penilaian. Terdapat 3 hal yang ditanyakan, yaitu mengenai antarmuka, level pada maze dan *fun*. Pada Tabel 5.3 ditunjukkan rata-rata hasil dari kuisisioner. Sedangkan kuisisioner dapat dilihat pada bagian lampiran kuisisioner. Kuisisioner diberikan kepada 5 orang yang telah mencoba perangkat lunak.

Tabel 5.3. Hasil Kuisioner

No.	Pernyataan	Rata –rata Penilaian
Penilaian Antarmuka		
1	Keindahan tampilan aplikasi	3,4
2	Antarmuka mudah untuk dioperasikan	3,6
Penilaian Level		
1	Level memberikan waktu adaptasi	3,2
2	Semakin besar level semakin menantang	3
Penilaian Fun		
1	Saya memiliki ketertarikan untuk bermain	3,4
2	Saya menikmati permainan	3,2
3	Saya ingin mencoba lagi permainan ini	3

Dari Tabel 5.3 dapat dilihat bahwa antarmuka sudah dapat diterima dengan baik oleh pemain. Lalu pembuatan level juga sudah bisa membuat pemain tertantang. Respon pemain untuk kategori *fun* sudah cukup bagus.

BAB VI

KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil selama proses perancangan, implementasi, serta pengujian dapat diambil kesimpulan sebagai berikut:

1. Perangkat lunak sudah dapat menerapkan skenario, level dan *gameplay roleplaying game* yang telah dirancang sebelumnya.
2. *Maze generator* pada perangkat lunak sudah dapat membuat maze yang berbeda dengan ukuran yang sama dan pintu keluar di setiap maze pasti bisa dicapai oleh pemain.
3. Antarmuka di dalam permainan sudah dapat diterima dengan baik oleh pemain, pembuatan level juga sudah bisa membuat pemain tertantang dan respon pemain untuk kategori fun sudah cukup bagus.

6.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang yaitu menambahkan cerita pada perangkat lunak sehingga perangkat lunak lebih menarik.

DAFTAR PUSTAKA

- 
- [1] N. Oxford, “RPG/Role-Playing Game,” About, [Online]. Available: <http://ds.about.com/od/glossary/g/Rpg-Role-Playing-Game.htm>. [Diakses 27 May 2015].
- [2] E. Jo, “Introduction to Mazes and Labyrinths,” Edkins Jo, [Online]. Available: <http://gwydir.demon.co.uk/jo/maze/intro/>. [Diakses 29 May 2015].
- [3] C. Corporation, “What Is a Maze?,” Conjecture Corporation, [Online]. Available: <http://www.wisegeek.com/what-is-a-maze.htm>. [Diakses 29 May 2015].
- [4] “The best development platform for creating games,” Unity, [Online]. Available: <https://unity3d.com/unity>. [Diakses 29 May 2015].
- [5] Blender, “About,” Blender, [Online]. Available: <http://www.blender.org/about/>. [Diakses 29 May 2015].
- [6] Jamis, “Maze Generation: Growing Tree algorithm,” Jamis, [Online]. Available: <http://weblog.jamisbuck.org/2011/1/27/maze-generation-growing-tree-algorithm>. [Diakses 29 May 2015].

BAB VI

KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil selama proses perancangan, implementasi, serta pengujian dapat diambil kesimpulan sebagai berikut:

1. Perangkat lunak sudah dapat menerapkan skenario, level dan *gameplay roleplaying game* yang telah dirancang sebelumnya.
2. *Maze generator* pada perangkat lunak sudah dapat membuat maze yang berbeda dengan ukuran yang sama dan pintu keluar di setiap maze pasti bisa dicapai oleh pemain.
3. Antarmuka di dalam permainan sudah dapat diterima dengan baik oleh pemain, pembuatan level juga sudah bisa membuat pemain tertantang dan respon pemain untuk kategori fun sudah cukup bagus.

6.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang yaitu menambahkan cerita pada perangkat lunak sehingga perangkat lunak lebih menarik.

LAMPIRAN POTONGAN KODE

```

if(_attack){
    _counterTimeAttack -= Time.deltaTime;
    if(_counterTimeAttack <= _delayAttack &&
    !_alreadyDamaged){
        if(allEnemyInRange.Count > 0){
            for(int i = 0; i <
allEnemyInRange.Count; i++){
                float dist = Vector3.Distance(
allEnemyInRange[i].transform.position,
_myTransform.position);
                if(_counterAttackAnimation >=
3){
                    if(dist <=
_playerCharacter.DistanceAttack * 2){
if(allEnemyInRange[i].StateEnemyNow !=
SimpleAI.StateEnemy.Attacked){
                        Weapon weaponNow =
PlayerCharacter.EquipWeapon.GetComponent<Weapon
>();
                        float tempDmg =
Random.Range(weaponNow.minAttack,
weaponNow.maxAttack);
                        tempDmg +=
_playerCharacter.allAttribute[3].Value;
allEnemyInRange[i].DoDamage(tempDmg);
                    }
                }else{
                    if(dist <=
_playerCharacter.DistanceAttack){
                        Vector3 dir2 =
(allEnemyInRange[i].transform.position
_myTransform.position).normalized;
                        float direction2 =

```

```

Vector3.Dot (dir2, _myTransform.forward);
                                if(direction2 >
_playerCharacter.MaxAngleAttack &&
allEnemyInRange[i].StateEnemyNow !=
SimpleAI.StateEnemy.Attacked){
                                Weapon weaponNow =
PlayerCharacter.EquipWeapon.GetComponent<Weapon
>();
                                float tempDmg =
Random.Range (weaponNow.minAttack,
weaponNow.maxAttack);
                                tempDmg +=
_playerCharacter.allAttribute[3].Value;
allEnemyInRange[i].DoDamage (tempDmg);
                                }
                                }
                                }
                                _alreadyDamaged = true;
                                }else{
                                _alreadyDamaged = true;
                                }
                                }else if(_counterTimeAttack <= 0){
                                _counterTimeAttack = TIME_ATTACK;
                                _attack = false;
                                _animator.SetBool ("Attack",false);
                                _delayAttack = TIME_ATTACK * 0.5f;
                                _alreadyDamaged = false;
                                }
                                }

```

Potongan Kode 1. Kode Menyerang

```

public void CreateAllCell(){
    Transform _newCellTemp;
    _allCell = new Transform[sizeX, sizeZ];
    for (int x = 0; x < sizeX; x++) {
        for (int z = 0; z < sizeZ; z++) {
            if((x == 0) && (z != (sizeZ - 1))){
                _newCellTemp = Instantiate
                (cellPrefeb2, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }else if((x == 0) && (z == (sizeZ -
1))){
                _newCellTemp = Instantiate
                (cellPrefeb1, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }else if((z == sizeZ - 1)){
                _newCellTemp = Instantiate
                (cellPrefeb3, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }else{
                _newCellTemp = Instantiate
                (cellPrefeb4, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }
            _newCellTemp.name = string.Format
            ("({0},0,{1})", x, z);
            _newCellTemp.parent = transform;
            _newCellTemp.GetComponent<Cell>().Position =
            new Vector3 (x, 0, z);
            _allCell[x, z] = _newCellTemp;
        }
    }
}

```

Potongan Kode 2. Fungsi CreateAllCell

```

private void FindNextCell() {
    Transform _nextCell;
    Cell _nextCellScript;

    int _nextCellPosX;
    int _nextCellPosZ;

    Transform _nowCell = visitedCellList[0];
    Cell _nowCellScript =
        visitedCellList[0].GetComponent<Cell> ();

    int _nowCellPosX =
        (int)_nowCellScript.Position.x;
    int _nowCellPosZ =
        (int)_nowCellScript.Position.z;

    int _randNumber = Random.Range(0, 4);
    int _counter = 0;
    Vector3 _randDirection;

    do{
        do{
            _randDirection =
                directions[_randNumber];
            _nextCellPosX = _nowCellPosX +
                (int)_randDirection.x;
            _nextCellPosZ = _nowCellPosZ +
                (int)_randDirection.z;
            _randNumber = (_randNumber + 1) % 4;
            _counter++;
            if (_counter > 4) {
                _nextMaze = _nextMaze + "-" +
                    "#";
                RemoveCompletedCellList(_nowCell);
                return;
            }
        }while(_nextCellPosX < 0 ||
            _nextCellPosZ < 0 || _nextCellPosX >= sizeX ||
            _nextCellPosZ >= sizeZ);
    }
}

```



```

        _nextCell = _allCell[_nextCellPosX,
        _nextCellPosZ];
        _nextCellScript = _nextCell.GetComponent
        <Cell> ();
        }while(_nextCellScript.AlreadyVisited);

        _nextMaze = _nextMaze + _nextCellPosX + ", "
        + _nextCellPosZ + "#";

        DestroyWalls(_nowCell, _nextCell);

        AddVisitedCellList(_nextCell);
    }

```

Potongan Kode 3. Fungsi FindNextCell

```

private void SetWhereBegin(int a, int b){
    AddVisitedCellList(_allCell[a, b]);
}

```

Potongan Kode 4. Fungsi SetWhereBegin

```

private void AddVisitedCellList(Transform
visitedCell){
    visitedCellList.Insert (0, visitedCell);

    Cell cs = visitedCell.GetComponent<Cell> ();
    cs.AlreadyVisited = true;
}

```

Potongan Kode 5. Fungsi AddVisitedCellList

```

private void RemoveCompletedCellList (Transform
cm){
    visitedCellList.Remove (cm);
}

```

Potongan Kode 6. Fungsi RemoveCompletedCellList

```

private void DestroyWalls(Transform a, Transform
b) {
    RaycastHit[] hits;

    Vector3 or = new Vector3(a.position.x,
a.position.y + heightRayCast, a.position.z);

    hits = Physics.RaycastAll(or, b.position -
a.position, longRayCast);

    Destroy(hits[0].transform.gameObject);
}

```

Potongan Kode 7. Fungsi DestroyWalls

```

public void CreateGrid ()
{
    int x = (int)GridSize.x;
    int z = (int)GridSize.z;
    int maxXZ = Mathf.Max (x, z);
    Camera.mainCamera.transform.position =
new Vector3 (maxXZ / 2f, maxXZ, maxXZ / 8f);
    GridArr = new Transform[x, z];
    Transform newCell;
    for (int ix = 0; ix < x; ix++) {
        for (int iz = 0; iz < z; iz++) {
            newCell = (Transform)Instantiate
(CellPrefab, new Vector3 (ix, 0, iz) * Buffer,
Quaternion.identity);
            newCell.name = string.Format
("{0},{1}", ix, iz);
            newCell.parent = transform;
            newCell.GetComponent<Cell>
().Position = new Vector3 (ix, 0, iz);
            GridArr [ix, iz] = newCell;
        }
    }
}

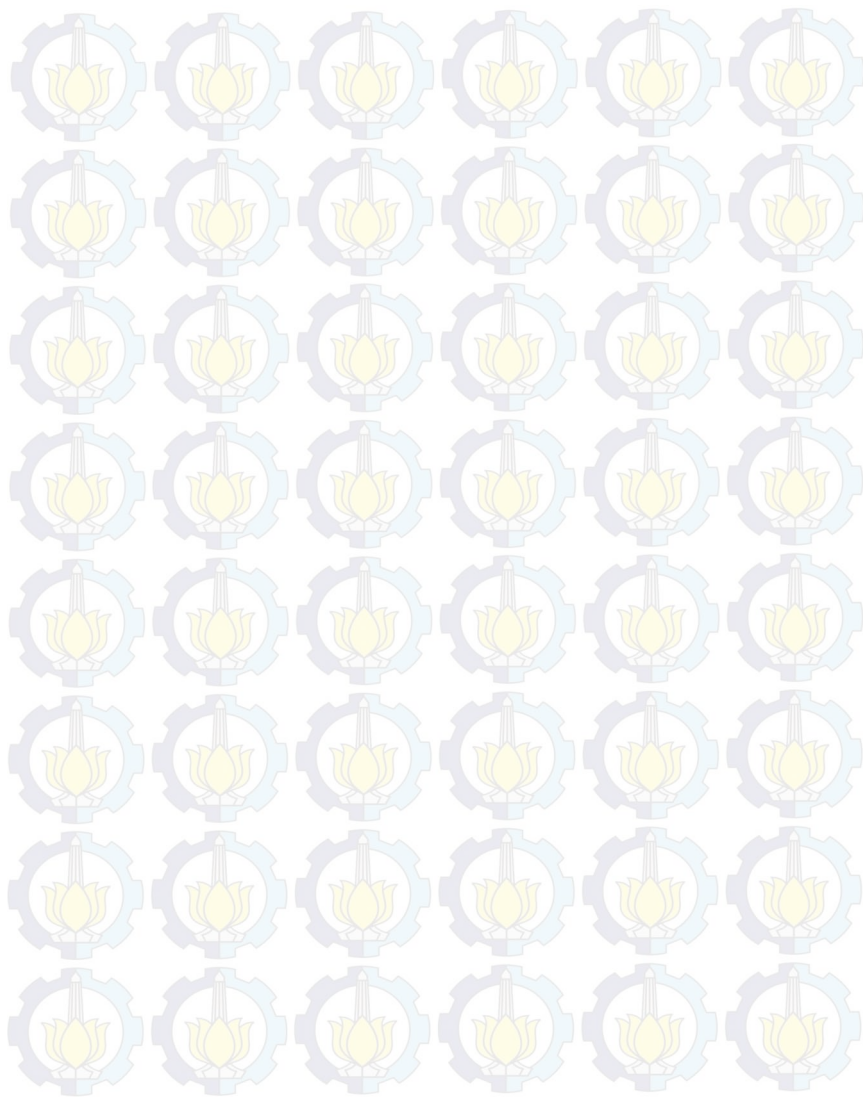
```

Potongan Kode 8. Potongan Kode Sel Dengan Satu Object

```
void Update ()
{
    if (Input.GetKeyDown (KeyCode.Tab) ||
    Input.GetKey (KeyCode.Space)) {
        InvokeRepeating("FindNext", 0,
        0.001f); //FindNext ();
    }
    if (Input.GetKeyDown
    (KeyCode.LeftShift)) {
        FindNext ();
    }
    if (Input.GetKeyDown (KeyCode.F1))
    {
        Application.LoadLevel(0);
    }
}
```

Potongan Kode 9. Pembuatan Maze Dengan Triger

[Halaman ini sengaja dikosongkan]



LAMPIRAN KUISIONER


KUISIONER THE WARRIOR

Nama Lengkap	M. Latif R. Kustamadi
Jenis Kelamin	Perempuan / Laki-Laki
Usia	22 tahun
Pekerjaan	Mahasiswa
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi				✓
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi				✓
2	Semakin besar level semakin menantang				✓
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain				✓
2	Saya menikmati permainan				✓
3	Saya ingin mencoba lagi permainan ini			✓	

Surabaya, 25 Juni 2015


 (-----)
 M. LATIF R.

KUISIONER THE WARRIOR

Nama Lengkap	MARANU TOTO NEGORO
Jenis Kelamin	Pemuaan / Laki-Laki
Usia	21 tahun
Pekerjaan	MAHASISWA
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi			✓	
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi			✓	
2	Semakin besar level semakin menantang			✓	
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain				✓
2	Saya menikmati permainan			✓	
3	Saya ingin mencoba lagi permainan ini				✓

Surabaya, 25 Juni 2015

(MARANU TOTO NEGORO)

KUISIONER THE WARRIOR

Nama Lengkap	Dzik Permana
Jenis Kelamin	Perempuan / Laki-Laki
Usia	22 tahun
Pekerjaan	Mahasiswa
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi				✓
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi				✓
2	Semakin besar level semakin menantang				✓
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain				✓
2	Saya menikmati permainan				✓
3	Saya ingin mencoba lagi permainan ini				✓

Surabaya, 25 Juni 2015

(Dzik Permana)
(.....)

KUISIONER THE WARRIOR

Nama Lengkap	Luthfan Arief Pratomo
Jenis Kelamin	Perempuan / Laki-Laki
Usia	21 tahun
Pekerjaan	Mahasiswa
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan, Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi			✓	
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi				✓
2	Semakin besar level semakin menantang				✓
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain			✓	
2	Saya menikmati permainan			✓	
3	Saya ingin mencoba lagi permainan ini				✓

Surabaya, 24 Juni 2015

(Luthfan A.R.)

KUISIONER THE WARRIOR

Nama Lengkap	Punggi Esthi Bawono
Jenis Kelamin	Perempuan / Laki-Laki
Usia	22 tahun
Pekerjaan	Mahasiswa
Instansi	Institut Teknologi Sepuluh Nopember

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi				✓
2	Antarmuka mudah untuk dioperasikan			✓	
Penilaian Level					
1	Level memberikan waktu adaptasi			✓	
2	Semakin besar level semakin menantang		✓		
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain			✓	
2	Saya menikmati permainan			✓	
3	Saya ingin mencoba lagi permainan ini			✓	

Surabaya, 25 Juni 2015

Punggi E.B
(.....)

Rancang Bangun Maze Generator untuk Roleplaying Game “The Warrior”

Nama Mahasiswa : I Ketut Megi Trisnawan
NRP : 5111100190
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing 1 : Imam Kuswardayan, S.Kom., M.T.
Dosen Pembimbing 2 : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

ABSTRAK

Dunia game saat ini berkembang dengan pesat. Banyak game baru yang bermunculan. Berbagai genre pun diusung game tersebut. Salah satu genre yang cukup populer adalah roleplaying game (RPG).

Roleplaying game adalah genre game dimana pemain diberikan kebebasan untuk mengatur karakter dalam pengaturan fiksi. Pemain juga diberikan kebebasan untuk melakukan suatu tindakan yang akan mempengaruhi jalan cerita pada game tersebut. Dalam perkembangannya kebanyakan game bergenre roleplaying game memiliki lingkungan/tempat bermain yang statis, artinya setiap dimainkan kembali lingkungan pemain di dalam game akan sama terus. Game The Warrior mencoba menghadirkan suatu yang berbeda dengan membuat lingkungan dalam permainan berbeda setiap dimainkan.

Game The Warrior mengambil lingkungan yang berbentuk maze. Di permulaan permainan akan dijalankan sebuah algoritma yang akan membuat maze secara acak namun bisa terpecahkan. Selain itu di dalam maze akan terdapat musuh-musuh yang harus dibunuh pemain sebelum memecahkan maze tersebut.

Kata Kunci: Maze Generator, Roleplaying Game

Design And Implementation Of Maze Generator For Roleplaying Game “The Warrior”

Student Name : I Ketut Megi Trisnawan
Student ID : 5111100190
Major : Teknik Informatika FTIf-ITS
Advisor 1 : Imam Kuswardayan, S.Kom., M.T.
Advisor 2 : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

ABSTRACT

Game world is currently growing rapidly. Many new games are popping up. Various genres also carried the game. One genre that is quite popular is the roleplaying game(RPG).

Roleplaying game is a genre of game where players are given the freedom of weeks to set up character in a fictional setting. Players are also given the freedom to perform an action that will affect the storyline in the game. In the development of most games genre role-playing game has environmental/playground is static, thats mean every every time the game played again the environment will still same like the first game played. Game The Warrior try a unique way to create different environments in the game every time that game played.

Game The Warrior took environments in maze. The beginning of the game will run an algorithm that will make the maze randomly but can be solved. Also in the maze there will be enemies who must be killed before the players solve the maze.

Keywords: Maze Generator, Roleplaying Game

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan tugas akhir yang berjudul :

“Rancang Bangun Maze Generator untuk Roleplaying Game “The Warrior””

Harapan dari penulis semoga apa yang tertulis di dalam buku tugas akhir ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini, serta dapat memberikan kontribusi yang nyata.

Dalam pelaksanaan dan pembuatan tugas akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak, tanpa mengurangi rasa hormat penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Tuhan Yang Maha Esa atas segala rahmat-Nya, sehingga penulis dapat menyelesaikan Tugas Akhir.
2. Bapak penulis, I Ketut Pinti, Ibu penulis, Ni Luh Mindi, dan keluarga yang selalu memberikan dukungan penuh untuk menyelesaikan tugas akhir ini.
3. Bapak Imam Kuswardayan dan Ibu Nanik Suciati selaku dosen pembimbing yang telah bersedia meluangkan waktu untuk memberikan petunjuk selama proses pengerjaan tugas akhir ini.
4. Bapak, Ibu dosen Jurusan Teknik Informatika ITS yang telah dengan sabar mendidik dan memberikan ilmu yang tak ternilai harganya bagi penulis.
5. Seluruh staf dan karyawan Teknik Informatika ITS yang banyak memberikan kelancaran administrasi akademik kepada penulis.
6. Ibet, Melfa, Kaspul, Fandi, Ajong dan Kemal terima kasih telah mengingatkan saya akan mantan.
7. Bayu yang memperbolehkan wifinya dipakai.

8. Teman-teman user TA Lab Interaksi, Grafika dan Seni yang telah memberikan banyak dukungan dan semangat kepada penulis.
9. Teman-teman angkatan 2011 jurusan Teknik Informatika ITS yang memberikan dorongan motivasi dan bantuan kepada penulis.
10. Serta pihak-pihak lain yang tidak dapat disebutkan satu persatu.

Penulis telah berusaha sebaik-baiknya dalam menyusun tugas akhir ini, namun penulis mohon maaf apabila terdapat kekurangan, kesalahan maupun kelalaian yang telah penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, Juni 2015

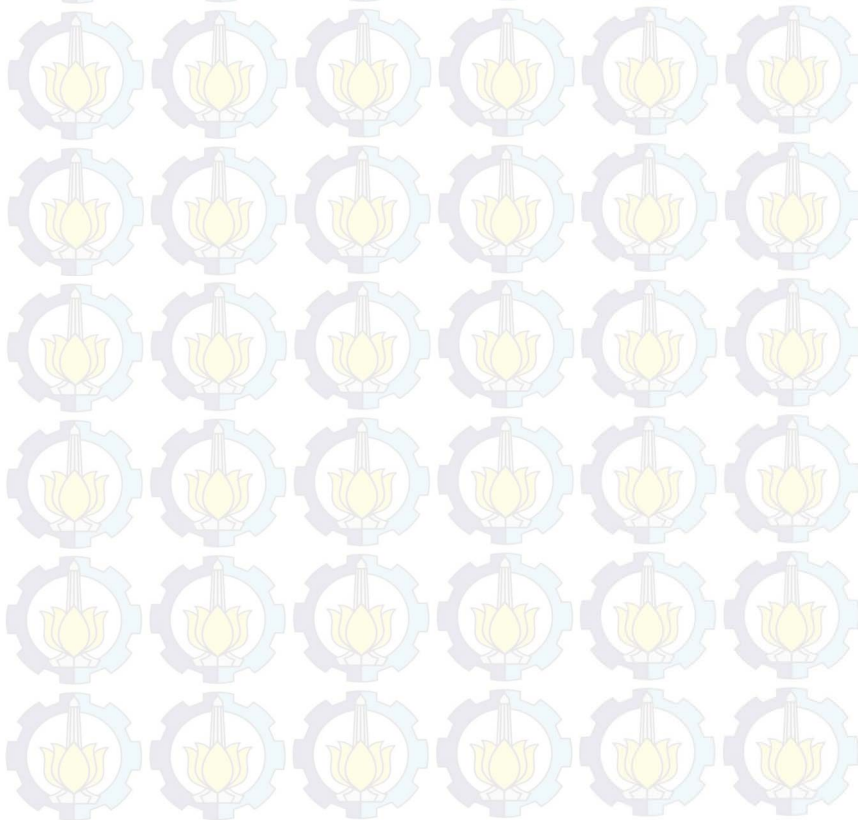
I Ketut Megi Trisnawan

DAFTAR ISI

LEMBAR PENGESAHAN	v
ABSTRAK	vii
ABSTRACT	ix
KATA PENGANTAR	xi
DAFTAR ISI	xiii
DAFTAR GAMBAR	xvii
DAFTAR TABEL	xxi
DAFTAR KODE SUMBER	xxiii
1 BAB I PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Tujuan	2
1.3. Rumusan Permasalahan	2
1.4. Batasan Permasalahan	2
1.5. Metodologi	2
1.6. Sistematika Penulisan	4
2 BAB II DASAR TEORI	7
2.1. <i>Roleplaying Game</i>	7
2.2. <i>Maze</i>	7
2.3. Unity	7
2.4. Blender	7
2.5. Algoritma <i>Growing Tree</i>	8
3 BAB III ANALISIS DAN PERANCANGAN SISTEM	9
3.1. Analisis Sistem	9
3.2. Perancangan Permainan	10
3.2.1. Deskripsi Umum Perangkat Lunak	10
3.2.2. Spesifikasi Kebutuhan Fungsional	11
3.2.3. Spesifikasi Kebutuhan Non-Fungsional	11
3.2.4. Karakteristik Pengguna	11
3.3. Perancangan Sistem	12
3.3.1. Kasus Penggunaan Menu Utama	12
3.3.2. Kasus Penggunaan Saat Dalam Permainan	18
3.3.3. Kontrol	21

3.3.4.	Antarmuka	21
3.3.5.	<i>Maze Generator</i>	26
3.3.6.	Level dan Skenario	33
3.4.	Design Asli dan Modifikasi <i>Growing Tree</i>	34
4	BAB IV IMPLEMENTASI	37
4.1.	Lingkungan Implementasi	37
4.2.	Implementasi Menu Utama	37
4.3.	Implementasi Penggunaan Saat Permainan	38
4.4.	Implementasi Kontrol	44
4.5.	Implementasi Antarmuka	45
4.5.1.	Antarmuka Menu Character Creation	45
4.5.2.	Antarmuka Character Window	45
4.5.3.	Antarmuka Inventory Window	48
4.5.4.	Antarmuka Loot Window	49
4.5.5.	Antarmuka Shop Window	49
4.5.6.	Antarmuka Dalam Pertarungan	50
4.6.	Implementasi <i>Maze Generator</i>	50
4.7.	Implementasi Level dan Skenario	52
4.8.	Pembuatan Karakter	53
5	BAB V PENGUJIAN DAN EVALUASI	57
5.1.	Lingkungan Uji Coba	57
5.2.	Skenario Uji Coba	57
5.3.	Hasil Uji Coba	59
5.3.1.	Menggunakan <i>Combo</i>	59
5.3.2.	Mengurangi Darah Musuh	60
5.3.3.	Mendapatkan <i>Gold</i>	60
5.3.4.	Pemain Mati	62
5.3.5.	<i>Pause</i>	62
5.3.6.	Memunculkan Character Window	63
5.3.7.	Memunculkan Inventory Window	64
5.3.8.	Membuka Peti	64
5.3.9.	Mengambil <i>Item</i>	65
5.3.10.	Membeli <i>Item</i>	66
5.3.11.	Pembuatan <i>Maze</i>	66
5.4.	Evaluasi	71

5.5. Kuisisioner	71
6 BAB VI KESIMPULAN DAN SARAN	73
6.1. Kesimpulan	73
6.2. Saran	73
Daftar Pustaka	75
Lampiran Potongan Kode	77
Lampiran Kuisisioner	85
BIODATA PENULIS	90



DAFTAR TABEL

Tabel 3.1. Karakteristik Pengguna	12
Tabel 3.2. Kode Kasus Penggunaan Menu Utama	13
Tabel 3.3. Melanjutkan Permainan	13
Tabel 3.4. Memulai Permainan Baru	14
Tabel 3.5. Melihat Kontrol	14
Tabel 3.6. Menutup Perangkat Lunak	15
Tabel 3.7. Level dan Skenario	34
Tabel 4.1. Lingkungan Implementasi	37
Tabel 5.1. Lingkungan Uji Coba	57
Tabel 5.2. Tabel Skenario Uji Coba	58
Tabel 5.3. Hasil Kuis	72

DAFTAR GAMBAR

Gambar 1 Diagram kasus Penggunaan Menu Utama	12
Gambar 2. Diagram Aktifitas Melanjutkan Permainan	15
Gambar 3. Diagram Aktifitas Memulai Permainan Baru	16
Gambar 4. Diagram Aktifitas Melihat Kontrol	17
Gambar 5. Diagram Aktifitas Menutup Perangkat Lunak	17
Gambar 6. FSM Pemain Menyerang	18
Gambar 7. FSM Membuka Peti	19
Gambar 8. FSM Character Window	19
Gambar 9. FSM Berbelanja	20
Gambar 10. FSM Keluar dari Maze	20
Gambar 11. Perancangan Menu Utama	22
Gambar 12. Perancangan Menu Creation	22
Gambar 13. Perancangan Character Window	23
Gambar 14. Perancangan Inventory Window	23
Gambar 15. Perancangan Loot Window	24
Gambar 16. Perancangan Antarmuka Shop Window	24
Gambar 17. Perancangan Antarmuka Dalam Pertarungan	25
Gambar 18. Kumpulan Sel Kubus	26
Gambar 19 Langkah 1 Algoritma Pada Sel Kubus 4x4	26
Gambar 20 Langkah 2 Algoritma Pada Sel Kubus 4x4	27
Gambar 21 Langkah 3 Algoritma Pada Sel Kubus 4x4	27
Gambar 22 Langkah 4 Algoritma Pada Sel Kubus 4x4	27
Gambar 23 Langkah 5 Algoritma Pada Sel Kubus 4x4	28
Gambar 24 Langkah 6 Algoritma Pada Sel Kubus 4x4	28
Gambar 25 Langkah 7 Algoritma Pada Sel Kubus 4x4	28
Gambar 26 Langkah 8 Algoritma Pada Sel Kubus 4x4	28
Gambar 27 Langkah 9 Algoritma Pada Sel Kubus 4x4	29
Gambar 28 Langkah 10 Algoritma Pada Sel Kubus 4x4	29
Gambar 29 Langkah 11 Algoritma Pada Sel Kubus 4x4	29
Gambar 30 Langkah 11 Algoritma Pada Sel Kubus 4x4	29
Gambar 31 Langkah 12 Algoritma Pada Sel Kubus 4x4	30
Gambar 32 Langkah 13 Algoritma Pada Sel Kubus 4x4	30

Gambar 33 Langkah 14 Algoritma Pada Sel Kubus 4x4.....	30
Gambar 34 Langkah 15 Algoritma Pada Sel Kubus 4x4.....	30
Gambar 35 Langkah 16 Algoritma Pada Sel Kubus 4x4.....	31
Gambar 36 Langkah 17 Algoritma Pada Sel Kubus 4x4.....	31
Gambar 37 Hasil Maze Dari Algoritma	31
Gambar 38 Maze 5x5	32
Gambar 39 Maze 6x6	32
Gambar 40 Maze 15x15	33
Gambar 41. Sel Pada Design Asli	35
Gambar 42. Semua Sel Pada Design Modifikas.....	35
Gambar 43. Tampilan Menu Utam.....	37
Gambar 44. Tampilan Character Creation.....	45
Gambar 45. Tampilan Character Window Tab Equipment.....	46
Gambar 46. Tampilan Tab Attributes Pada Character Window.....	46
Gambar 47. Tampilan Point Attribute Jika Tidak Nol.....	47
Gambar 48. Tampilan Tab Sklills Pada Character Window.....	47
Gambar 49. Tampilan Jika Point Skill Tidak Nol	48
Gambar 50. Tampilan Inventory Window.....	48
Gambar 51. Tampilan Loot Window.....	49
Gambar 52. Tampilan Shop Window	49
Gambar 53. Tampilan Dalam Pertarungan.....	50
Gambar 54. Screenshoot dari Unity Untuk Level 1-11	52
Gambar 55. Implementasi Level 12-20	53
Gambar 56. Karakter Utama.....	54
Gambar 57. Peti.....	54
Gambar 58. Monster.....	55
Gambar 59. All Item.....	55
Gambar 60. Semua Serangan <i>Combo</i>	59
Gambar 61. Darah Musuh Berkurang.....	60
Gambar 62. Gold Pemain 0	61
Gambar 63. Gold Pemain Bertambah.....	61
Gambar 64. Pemain Mati.....	62
Gambar 65. Pause.....	63
Gambar 66. Character Window.....	63
Gambar 67. Inventory Window.....	64

Gambar 68. Membuka Peti.....	65
Gambar 69. Mengambil <i>Item</i> dari Loot Window	65
Gambar 70. Membeli Item	66
Gambar 71. Hasil 1 Maze 6x6.....	67
Gambar 72. Hasil 2 Maze 6x6.....	67
Gambar 73. Hasil 3 Maze 6x6.....	68
Gambar 74. Hasil 1 Maze 10x10.....	68
Gambar 75. Hasil 2 Maze 10x10.....	69
Gambar 76. Hasil 3 Maze 10x10.....	69
Gambar 77. Hasil 1 Maze 14x14.....	70
Gambar 78. Hasil 2 Maze 14x14.....	70
Gambar 79. Hasil 3 Maze 14x14.....	71

BIODATA PENULIS



Penulis, I Ketut Megi Trisnawan, lahir di kota Amlapura pada tanggal 29 Mei 1992. Penulis adalah anak keempat dari empat bersaudara dan dibesarkan di kota Amlapura, Bali.

Penulis menempuh pendidikan formal di SDN 1 Karangasem(1999-2005), SMPN 2 Amlapura (2005-2008), SMAN 2 Amlapura (2008-2011). Pada tahun 2011, penulis memulai pendidikan S1 jurusan Teknik Informatika Fakultas Teknologi Informasi di Institut Teknologi Sepuluh Nopember Surabaya, Jawa Timur.

Di jurusan Teknik Informatika, penulis mengambil bidang minat Interaksi, Grafika dan Seni dan memiliki ketertarikan di bidang Pembuatan Game. Walaupun penulis belum pernah mengukir prestasi untuk kampus tercinta bukan berarti penulis tidak pernah berusaha. Penulis sudah mencoba dengan segala kemampuan namun apadaya keberuntungan tidak menghampiri penulis. Dan penulis pernah menjadi asisten Animasi Komputer dan Pemodelan 3D. Penulis dapat dihubungi melalui alamat email ketut.3snawan@gmail.com.

BAB I

PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan tugas akhir, dan sistematika penulisan.

1.1. Latar Belakang

Saat ini *video game* berkembang dengan pesat. Berbagai ide dituangkan menjadi sebuah *video game* yang kreatif dengan berbagai *genre* yang diusungnya. *Video game* menjadi suatu lahan industri yang sangat menjanjikan di masa depan. Bahkan saat ini banyak orang yang sehari-harinya hanya bermain *video game* namun bisa menghasilkan uang yang cukup banyak.

Berhasilnya suatu *video game* dipengaruhi berbagai faktor. Salah satunya lingkungan/*environment* yang menjadi tempat bermain karakter pemain. Sering kali pemain merasa cepat bosan karena lingkungan *video game* hanya itu-itu saja. Setiap berganti level keadaan lingkungan *video game* tidak berubah. Untuk membuat sebuah lingkungan yang berubah-ubah setiap level secara manual tentu memakan waktu yang cukup lama. Maka diperlukan pembuat lingkungan secara otomatis. Berbagai tema untuk lingkungan bisa dibuat dinamis, dan di tugas akhir ini tema yang akan diusung adalah *maze*.

Faktor lainnya adalah *genre* yang diusung oleh *video game* tersebut. Di dunia *video game* mengenal berbagai macam *genre* yang memiliki peminatnya sendiri-sendiri. Salah satu *genre* yang cukup banyak peminatnya adalah *roleplaying game*. Di dalam *genre* ini pemain diberikan kebebasan untuk mengatur karakter dalam pengaturan fiksi.

Tugas akhir ini akan mencoba memadukan salah satu *genre* yaitu *roleplaying game* dengan lingkungan *maze* yang dinamis, dimana setiap dimainkan *maze* akan berubah. Dengan

dinamisnya lingkungan ini diharapkan pemain tidak akan cepat merasa bosan.

1.2. Tujuan

Tujuan dari pembuatan tugas akhir ini adalah menerapkan algoritma *growing tree* untuk membuat *maze* generator pada game The Warrior dengan *genre roleplaying game*.

1.3. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana merancang dan membuat *scenario*, level dan *gameplay roleplaying game* pada *video game* The Warriror?
2. Bagaimana membuat rancangan *maze* yang sesuai dengan aturan permainan dan memungkinkan untuk diselesaikan?
3. Bagaimana membuat dan menerapkan algoritma *growing tree* untuk membuat *maze* generator pada game The Warrior?

1.4. Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Pembuatan *video game* menggunakan Unity dan bahasa pemrograman C#.
2. *Maze* yang akan dibuat termasuk visualisasi dan *scenario*.
3. Dalam permulaan permainan *maze* akan di *generate* (*maze* akan di *generate* ulang apabila pemain memilih New Game pada *menu*).

1.5. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan tugas akhir ini yaitu:

1. Studi literatur

Pada tahap ini merupakan tahap pengumpulan dan pembelajaran informasi yang akan digunakan untuk

mengimplementasikan Tugas Akhir. Literatur yang digunakan adalah sebagai berikut:

- a. *Roleplaying game*;
- b. Unity;
- c. Maze;
- d. Algoritma *Growing Tree*;

2. Analisis dan Perancangan Sistem

Pada tahap ini, akan dilakukan analisa, perancangan dan pendefinisian kebutuhan sistem untuk mengetahui permasalahan yang akan dihadapi pada tahap implementasi. Kemudian, akan dijabarkan kebutuhan-kebutuhan tersebut kedalam perancangan fitur sistem. Berikut langkah yang akan dilakukan perancangan proses aplikasi;

- a. Perancangan *maze generator*;
- b. Perancangan *gameplay*;
- c. Perancangan data dan *asset game*;
- d. Perancangan menu.

3. Implementasi

Pada tahap ini dilakukan pembuatan aplikasi permainan beserta sistem yang terkait sesuai dengan tahap sebelumnya. Aplikasi akan dibuat dengan bantuan *game engine* Unity dengan menggunakan bahasa pemrograman C#.

4. Pengujian dan evaluasi

Pada tahap ini akan dilakukan pengujian terhadap perangkat lunak menggunakan beberapa metode yaitu pengujian *blackbox*. Pengujian *blackbox* adalah metode pengujian yang berfokus pada fungsionalitas tanpa melihat internal kode untuk mengetahui kemungkinan *bug* dan error. Pengujian dilakukan dengan berbagai test case yang memungkinkan terjadinya error dan *bug*.

5. Penyusunan buku tugas akhir

Pada tahap ini dilakukan penyusunan laporan dari seluruh konsep, dasar teori, implementasi, proses yang telah dilakukan, dan hasil-hasil yang telah didapatkan selama pengerjaan tugas akhir.

1.6. Sistematika Penulisan

Buku tugas akhir ini bertujuan untuk memberikan gambaran dan dokumentasi dari pengerjaan tugas akhir yang dilakukan. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut.

Secara garis besar, buku tugas akhir ini terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan tugas akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan tugas akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai analisis perangkat lunak meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifikasi pengguna. Kemudian perancangan sistem meliputi perancangan *maze* generator, *gameplay*, data beserta *asset*, dan antarmuka.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan perangkat lunak.

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian terhadap perangkat lunak

yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi mengetahui kemampuan perangkat lunak.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan dan saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan tugas akhir.

Lampiran

Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini.

BAB II

DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan tugas akhir ini. Teori-teori tersebut meliputi .

2.1. *Roleplaying Game*

Roleplaying game merupakan *genre game* dimana pemainnya bisa mengatur karakter dalam pengaturan fiksi. Seiring dengan berjalannya waktu pengertian *roleplaying game* tidak terbatas hanya pada seperangkat pengaturan fiksi, namun juga cerita di dalamnya, sistem pertarungannya dan sistem pertumbuhan karakter pemain yang memungkinkan karakter pemain bertambah kuat seiring dengan permainan tersebut dimainkan [1].

2.2. *Maze*

Banyak orang mengartikan *maze* adalah tempat dimana terdapat dinding tinggi atau tepian, dimana kita bisa berkeliling dan tersesat [2]. *Maze* juga berarti struktur yang kompleks dengan serangkain jalur interkoneksi. Istilah ini juga digunakan untuk merujuk kepada teka-teki grafis yang mereplikasi *maze* pada media dua dimensi. Untuk memecahkan teka-teki ini pemain harus mencari jalan ke pintu keluar atau lokasi lain [3].

2.3. *Unity*

Unity adalah sebuah platform pengembangan yang fleksibel dan kuat untuk menciptakan 3D dan 2D game multiplatform dan pengalaman interaktif. Unity merupakan platform yang lengkap bagi siapa saja yang bertujuan untuk membangun bisnis perangkat lunak yang berkualitas dan menghubungkan para pemain [4].

2.4. *Blender*

Blender adalah sebuah perangkat lunak yang gratis dan *open source* untuk animasi 3D. Mendukung keseluruhan dari 3D,

pipeline—modeling, rigging, animation, simulation, rendering, compositing dan motion tracking, bahkan video editing dan game creation. Blender adalah cross-platform dan berjalan dengan baik pada Linux, Windows dan Macintosh computer [5].

2.5. Algoritma Growing Tree

Tree merupakan salah satu contoh dari data struktur yang sering digunakan pada *computer science*. Data struktur ini memiliki *root*, *branches* dan *leaves*. Dalam penggunaannya terdapat beberapa cara antara lain *breadth first traversal* dan *depth frist traversal*. Algoritma yang akan digunakan pada tugas akhir ini yaitu *growing tree* menerapkan *depth frist traversal*.

Pada algoritma ini pertama-tama dibangun kumpulan sel kubus(tanpa bagian atas). Lalu dilakukan algoritma seperti di bawah ini:

1. Pilih secara acak satu sel, dan letakkan pada list.
2. Pilih satu sel pada list, pilih secara random tetangga sel yang belum dikunjungi, tambahkan tetangga tersebut pada list. Jika tidak ada tetangga yang bisa dikunjungi lagi hapus sel dari list.
3. Ulangi langkah 2 sampai list tersebut kosong [6].

BAB III

ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis sistem secara umum yang akan dibangun pada perangkat lunak ini. Selanjutnya dibahas mengenai perancangan permainan yang dibuat. Perancangan direpresentasikan dengan diagram UML (*Unified Modelling Language*) dan FSM (*Final State Machine*).

3.1. Analisis Sistem

Video game mempunyai berbagai macam *genre*, salah satunya adalah *roleplaying game*. *Roleplaying game* memiliki kelebihan tersendiri sehingga bila dimainkan tidak akan membuat bosan pemain, kelebihan tersebut adalah dengan membiarkan pemain mengatur karakter dalam pengaturan fiksi. Dalam pengaturan fiksi ini pemain biasanya akan diberikan *point* saat berhasil menaikkan level karakter, lalu *point* tersebut bisa digunakan untuk meningkatkan salah satu *attribute* dari karakter yang dimainkan oleh pemain. Misalkan saja jika dalam karakter yang dimainkan pemain terdapat *health*/darah, maka pemain bisa meningkatkan jumlah *health*/darah dengan menambahkan *point* yang telah diberikan kepada pemain. Sehingga pemain bisa mewujudkan karakter bagaimana yang ingin digunakan pada permainan, apakah karakter dengan darah yang banyak atau dengan *attack* yang besar.

Namun jika pemain mengulang kembali permainan dari awal pemain mungkin akan merasa bosan karena pemain akan kembali bermain pada lingkungan yang sama. Untuk menghindari ini maka akan dibangun *maze* yang akan berubah jika permainan diulang dari awal, diharapkan dengan lingkungan yang dinamis pemain tidak cepat merasa bosan karena permainan akan menjadi lebih menantang dan menarik.

Penulis membangun *video game* ini untuk *computer/desktop* dan menggunakan *game engine* Unity 3D dengan bahasa pemrograman C#. *Video game* ini mengambil

genre roleplaying game dan dibangun dengan *maze generator* yang bisa membangun *maze* yang variatif sehingga bisa membuat pemain betah untuk memainkan permainan ini.

3.2. Perancangan Permainan

3.2.1. Deskripsi Umum Perangkat Lunak

Video game yang akan dikembangkan pada tugas akhir ini adalah sebuah *video game* 3D dengan *genre roleplaying game* dengan lingkungan tempat bermain mengambil tema *maze*. Di dalam permainan ini pemain akan mengendalikan satu karakter yang memiliki beberapa *attribute* dan bisa memakai beberapa *item*. Untuk menyelesaikan permainan ini pemain harus bisa keluar dari *maze* di setiap level permainannya.

Karakter yang dikendalikan pemain akan memiliki beberapa *attribute* yaitu, *health* mempresentasikan jumlah darah pemain(jika darah lebih kecil atau sama dengan 0 maka pemain dianggap mati/kalah), *mana* untuk mempresentasikan jumlah *mana* maksimal yang dimiliki pemain(*mana* ini berguna untuk mengeluarkan *magic/skill*), *attack* untuk mempresentasikan jumlah *attack*(jumlah *damage* yang akan dikeluarkan pemain saat menyerang musuh), dan *defense* untuk mempresentasikan jumlah *defense* yang akan berguna untuk menahan serangan dari musuh.

Di dalam permainan terdapat 2 jenis *item*, yaitu *item* yang bisa dikonsumsi dan *item* yang tidak bisa dikonsumsi. *Item* yang bisa dikonsumsi seperti *health potion* yang apabila dikonsumsi akan menambah darah pemain, *mana potion* akan menambah *mana* bila digunakan dan *experient bonus* akan menambah jumlah *experient*, *experience* ini berguna untuk menaikkan level karakter. Selanjutnya *item* yang tidak bisa dikonsumsi terdiri dari senjata yang apabila dipakai dapat menaikkan *attack* dari karakter dan *item* yang bisa menaikkan *defense* yaitu *item* pelindung kepala, *item* pelindung kaki, *item* pelindung tangan dan *item* pelindung dada.

Di dalam permainan akan terdapat beberapa level yang harus dilewati agar menyelesaikan permainan ini. Disetiap levelnya akan terdapat *maze* dengan luas yang berbeda-beda. *Maze* ini akan dibuat dengan *maze* generator yang bisa membuat *maze* secara acak. Jika pemain memulai permainan baru, *maze* generator akan membuat *maze* secara acak dan menghasilkan *maze* yang berbeda dengan sebelumnya. Untuk keluar dari *maze* ini pemain harus mengumpulkan *key*/kunci yang terdapat pada peti yang penempatannya juga diacak pada permainan. Setelah mengumpulkan semua kunci pemain harus menuju pintu keluar yang penempatannya juga di *generate* secara acak di dalam *maze*.

3.2.2. Spesifikasi Kebutuhan Fungsional

Kebutuhan fungsional untuk *video game* ini hanya satu yaitu bisa dimainkan oleh pengguna.

3.2.3. Spesifikasi Kebutuhan Non-Fungsional

1. *FrameRate*

Framerate yang bagus merupakan kebutuhan standar untuk semua *video game*. Karena jika *framerate* terlalu kecil akan menyebabkan permainan melambat/*lag*, sehingga mengganggu kenyamanan permainan.

2. Grafis

Dikarenakan *video game* ini dibangun pada 3D dan *maze* yang dibangun terdiri dari banyak objek dibutuhkan grafis yang mencukupi agar permainan tidak melambat/*lag* yang bisa mengganggu kenyamanan memainkan *video game* ini.

3.2.4. Karakteristik Pengguna

Karakteristik pengguna tercantum pada Tabel 3.1. Hak akses yang akan diberikan kepada pemain hanya memainkan permainan, pemain tidak akan diberi hak akses untuk mengubah konten dari permainan. Di dalam permainan tidak tersedia bagaimana cara memainkan permainan ini, namun akan pemain dapat melihat tombola pa saja yang bisa dipakai dalam permainan ini.

Tabel 3.1. Karakteristik Pengguna

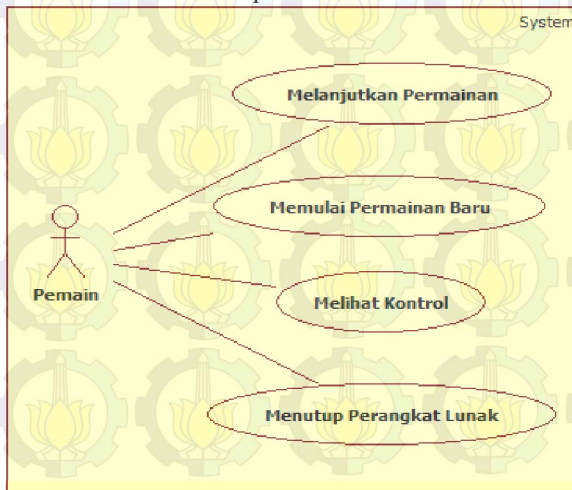
Nama Aktor	Tugas	Hak Akses Perangkat Lunak	Kemampuan yang Harus Dimiliki
Pemain	Memainkan permainan	Memainkan permainan	Mengerti bahasa Inggris dasar, pernah bermain <i>video game</i> dengan <i>genre roleplaying game</i> (tidak harus, tetapi sangat disarankan)

3.3. Perancangan Sistem

Perancangan sistem dibagi menjadi beberapa bagian, yaitu kasus penggunaan menu utama, kasus penggunaan dalam permainan, *control*, antarmuka dan *maze* generator.

3.3.1. Kasus Penggunaan Menu Utama

Bagian ini akan menjelaskan perancangan saat pengguna berada pada menu utama. Kasus penggunaan saat menu utama dicantumkan pada Gambar 1.



Gambar 1 Diagram kasus Penggunaan Menu Utama

Tabel 3.2. Kode Kasus Penggunaan Menu Utama

No	Kode Kasus Penggunaan	Nama Kasus Penggunaan	Keterangan
1	UC-001	Melanjutkan permainan	Untuk melanjutkan permainan dari data yang tersimpan terakhir
2	UC-002	Memulai permainan baru	Untuk memulai permainan dari awal
3	UC-003	Melihat control	Untuk melihat kontrol yang bisa dipakai dalam permainan
4	UC-004	Menutup Perangkat Lunak	Untuk menutup perangkat lunak

3.3.1.1. Spesifikasi Kasus Penggunaan Menu Utama

Berikut merupakan penjelasan kasus penggunaan menu utama. Terdapat 4 kasus penggunaan dalam menu utama, yaitu melanjutkan permainan, memulai permainan baru, melihat control dan menutup perangkat lunak.

Tabel 3.3. Melanjutkan Permainan

Nama	Melanjutkan permainan
Kode	UC-001
Deskripsi	Melanjutkan permainan dari data yang terakhir disimpan
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 1. Pemain menekan tombol Load Game 2. Sistem mengambil data yang tersimpan dan membangkitkan <i>scene</i> sesuai data yang tersimpan 3. Selesai

Tabel 3.4. Memulai Permainan Baru

Nama	Memulai permainan baru
Kode	UC-002
Deskripsi	Memilih menu permainan baru
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 1. Pemain menekan tombol New Game 2. Sistem menghapus data lama 3. Sistem menampilkan scene Creating New Character 4. Pemain mengisi persyaratan yaitu, mengisi nama karakter, memakai <i>point attribute</i> dan memilih <i>element</i> 5. Sistem memeriksa semua persyaratan sudah terisi. Sistem menampilkan tombol Lets Begin 6. Selesai <p>A.1. Pemain tidak mengisi persyaratan dengan lengkap</p>
Alur Alternatif	<p>A.1. Pemain tidak mengisi persyaratan dengan lengkap</p> <p>A.1.1. Kembali ke alur 3</p>

Tabel 3.5. Melihat Kontrol

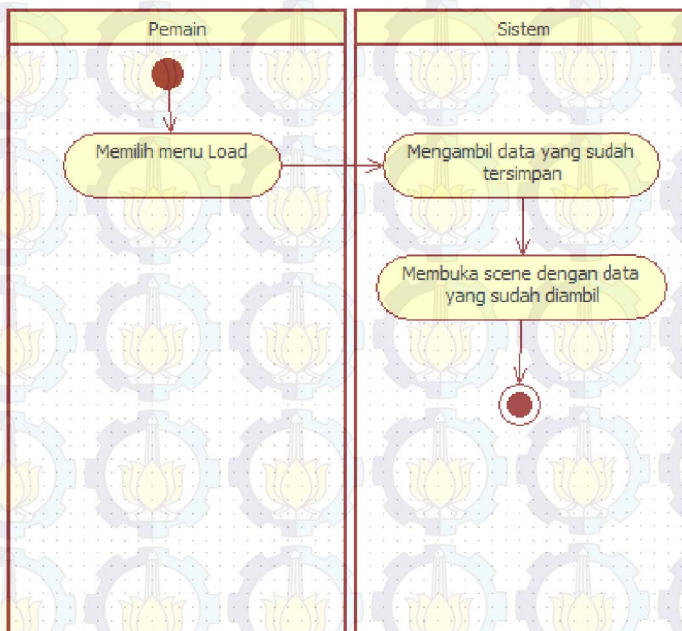
Nama	Melihat kontrol
Kode	UC-003
Deskripsi	Melihat tombol apa saja yang bisa dipakai pada permainan
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 4. Pemain menekan tombol Control 5. Sistem menampilkan scene Control 6. Selesai

Tabel 3.6. Menutup Perangkat Lunak

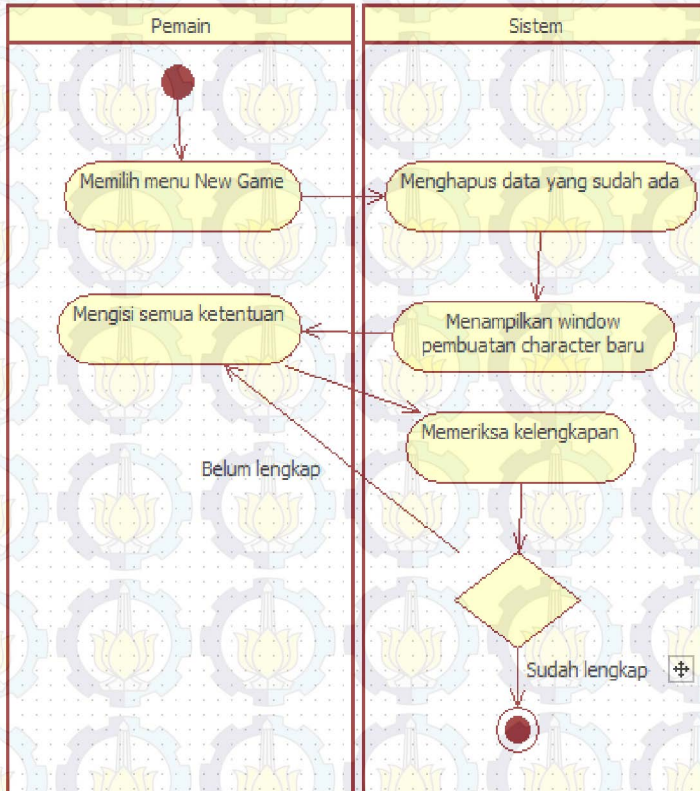
Nama	Menutup Perangkat lunak
Kode	UC-004
Deskripsi	Menutup perangkat lunak
Aktor	Pemain
Kondisi Awal	Pemain telah berada di scene menu utama
Alur Normal	<ol style="list-style-type: none"> 1. Pemain menekan tombol Exit 2. Sistem menampilkan scene Control 3. Selsesai

3.3.1.2. Diagram Aktifitas Penggunaan Menu Utama

Berikut merupakan diagram aktifitas dari kasus penggunaan menu utama.

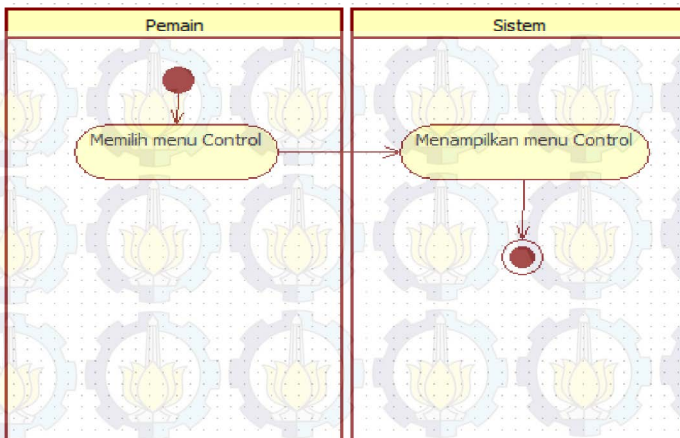
**Gambar 2. Diagram Aktifitas Melanjutkan Permainan**

Gambar 2 menunjukkan diagram aktifitas untuk melanjutkan permainan. Saat pemain memilih menu Load, sistem akan membaca data yang telah tersimpan lalu membuat scene berdasarkan data tersebut.



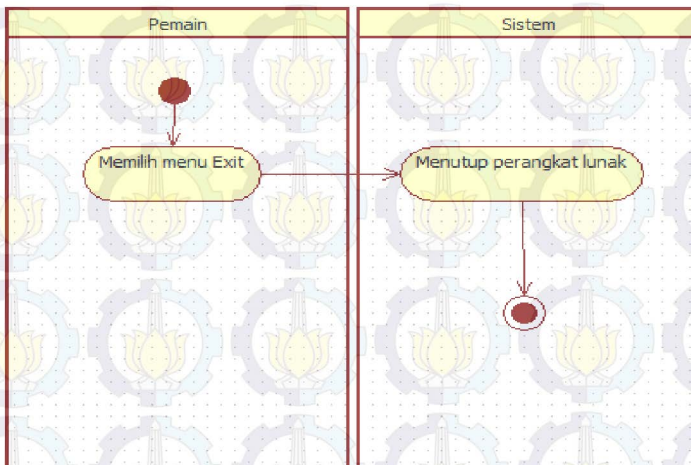
Gambar 3. Diagram Aktifitas Memulai Permainan Baru

Gambar 3 menunjukkan diagram aktifitas saat pemain memilih menu New Game. Sistem akan menghapus data yang sudah. Kelengkapan yang dimaksud adalah mengisi nama dan memakai *point attribute*. Pemain harus mengisi kelengkapan terbut, jika kelengkapan ini tidak terpenuhi pemain tidak akan bisa melanjutkan ke tahap selanjutnya.



Gambar 4. Diagram Aktivitas Melihat Kontrol

Gambar 4 diagram aktivitas saat pemain memilih menu Control.

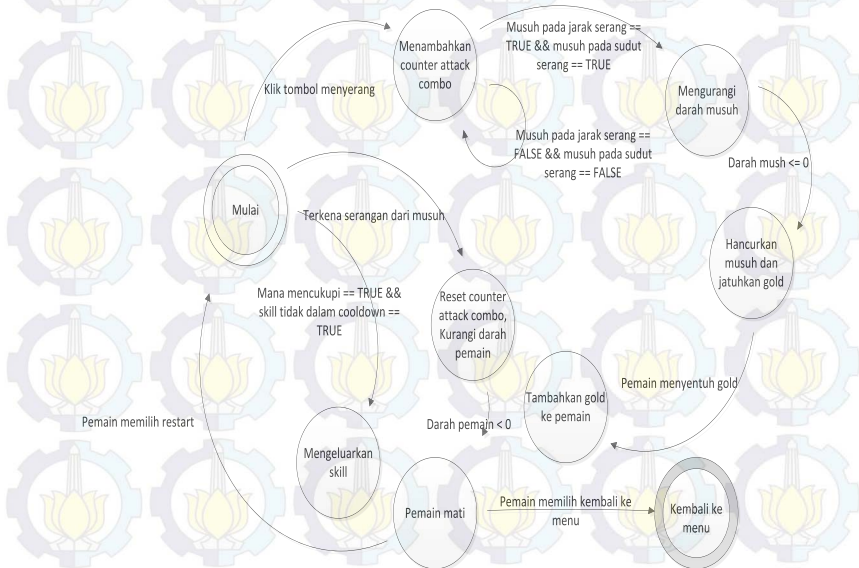


Gambar 5. Diagram Aktivitas Menutup Perangkat Lunak

Gambar 5 merupakan diagram aktivitas saat pemain ingin keluar dari permainan.

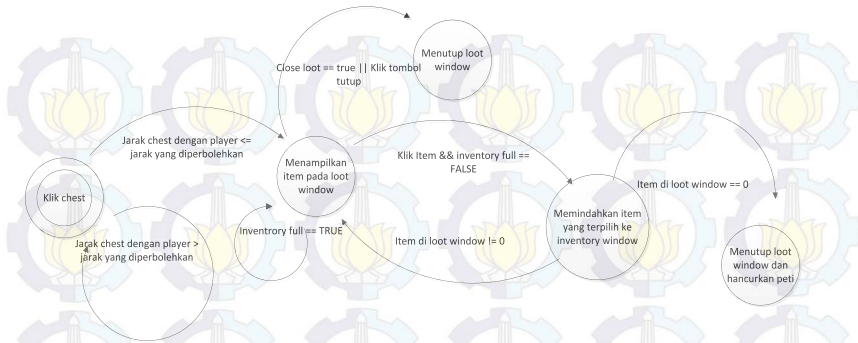
3.3.2. Kasus Penggunaan Saat Dalam Permainan

Dalam perancangan kasus penggunaan dalam permainan akan dijelaskan dengan FSM.



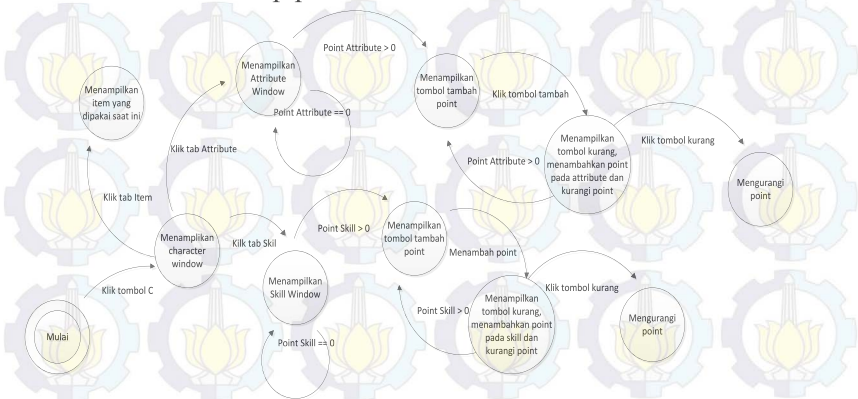
Gambar 6. FSM Pemain Menyerang

Gambar 6 menjelaskan saat pemain menekan tombol serang. Saat pemain menekan tombol serang maka *counter* untuk *attack combo* bertambah dan jika ada musuh dalam jangkauan serang maka kurangi darah musuh tersebut. Jika darah musuh kurang dari atau sama dengan 0 maka hancurkan musuh dan jatuhkan *gold* untuk pemain. *Gold* akan ditambahkan ke pemain jika pemain menyentuh *gold* tersebut. Namun jika saat menyerang pemain terkena serangan dari musuh maka *counter attack combo* direset menjadi 0. Pemain bisa mengeluarkan skill jika mana mencukupi dan *skill* tidak sedang *cooldown*.



Gambar 7. FSM Membuka Peti

Pada Gambar 7 menjelaskan saat pemain membuka peti. Peti hanya bisa dibuka bila jarak pemain dengan peti tidak melebihi yang telah ditentukan. Saat pemain membuka peti, peti akan mengacak jumlah *item* yang akan diberikan. *Item* akan ditampilkan di *loot window*. Pemain bisa mengambilnya dengan klik *item* pada loot Window jika *inventory* pemain tidak penuh. Peti akan tertutup secara otomatis jika *item* pada 0 atau pemain jarak menjauhi peti. Pemain juga bisa menutup peti dengan menekan tombol tutup pada *loot window* nanti.



Gambar 8. FSM Character Window

FSM pada Gambar 8 menjelaskan alur saat pemain membuka character window. Pada character window terdapat 3 tab, yaitu tab *item*, *attribute* dan *skill*. Pada tab *item* terdapat *item* apa saja yang sedang dipakai oleh pemain. Lalu tab *attribute* digunakan untuk melihat dan memperbarui status *attribute* pemain. Yang terakhir tab *skill* untuk melihat dan memperbarui *skill point* pemain.



Gambar 9. FSM Berbelanja

Pada Gambar 9 menjelaskan saat pemain akan berbelanja. Pemain harus berada pada jarak tertentu untuk dapat membuka *window* berbelanja. Untuk berbelanja pemain harus menukarkan gold sesuai dengan harga barang yang ingin dibeli. Dan jika *inventory* pemain tidak penuh.



Gambar 10. FSM Keluar dari Maze

Gambar 10 menjelaskan saat pemain keluar dari *maze*. Pemain akan keluar dari maze dan melanjutkan permainan apabila pemain telah mengumpulkan semua kunci yang akan tersembunyi di salah satu peti dan melewati pintu keluar.

3.3.3. Kontrol

Di dalam permainan ini terdapat beberapa tombol yang bisa digunakan yaitu:

1. Tombol W untuk menggerakkan karakter maju.
2. Tombol S untuk menggerakkan karakter mundur/bergerak ke belakang.
3. Tombol A dan D untuk berputar.
4. Tombol C untuk memunculkan dan menyembunyikan karakter *window*.
5. Tombol I untuk memunculkan dan menyembunyikan *inventory window*.
6. Tombol P untuk memberhentikan sementara permainan.
7. Tombol 1 dan 2 untuk menggunakan *skill*.
8. Tombol kanan mouse untuk menyerang.
9. Tombol kiri mouse untuk membuka peti, mengambil barang, membeli *item* dan menggunakan *item*.
10. *Double click* tombol kiri mouse untuk memakai item.

3.3.4. Antarmuka

Dalam subbab ini diperlihatkan beberapa rancangan antarmuka dalam permainan.

3.3.4.1. Antarmuka Menu Utama

Pada Gambar 11 menunjukkan rancangan pada menu utama pada permainan. Terdapat 4 tombol yaitu *Load* untuk melanjutkan permainan dari data yang tersimpan terakhir. Untuk memulai permainan baru. *New Game* untuk memulai permainan baru. *Control* untuk melihat tombol apa saja yang bisa dipakai dalam permainan. *Exit* untuk keluar dari permainan.



Gambar 11. Perancangan Menu Utama

3.3.4.2. Antarmuka Menu Creation

 A rectangular window for creating a new game. It contains the following elements:

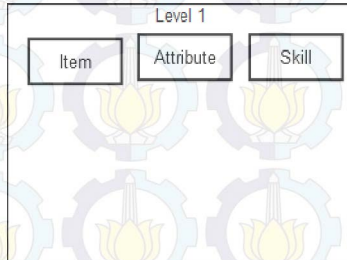
- Name :** A text input field.
- Point Left : 10** (Static text)
- Health : 10** (Static text) followed by a minus sign button '-' and a plus sign button '+'.
- Health : 10** (Static text) followed by a minus sign button '-' and a plus sign button '+'.
- Health : 10** (Static text) followed by a minus sign button '-' and a plus sign button '+'.
- Health : 10** (Static text) followed by a minus sign button '-' and a plus sign button '+'.
- Element :** Two buttons labeled 'Fire' and 'Earth'.
- Skill :** Two buttons labeled 'Skill1' and 'Skill2'.
- Creating** (A button at the bottom right).

Gambar 12. Perancangan Menu Creation

Pada Gambar 12 merupakan rancangan Menu *Creation*. Menu ini akan muncul setelah pemain memilih menu *New Game* .

Pada menu ini pemain harus mengisi nama, menggunakan poin dan memilih element untuk melanjutkan permainan.

3.3.4.3. Antarmuka Character Window



Gambar 13. Perancangan Character Window

Gambar 13 menunjukkan Character Window, terdapat 3 tab yaitu tab Item untuk melihat item apa saja yang sedang dipakai oleh pemain. Tab Attribute untuk melihat *attribute* pemain saat ini, di dalam tab ini bisa menambahkan *point attribute* jika pemain mempunyai *point attribute*. Tab skill untuk melihat dan menambahkan *point skill* jika pemain mempunyai *point skill*.

3.3.4.4. Antarmuka Inventory Window



Gambar 14. Perancangan Inventory Window

Pada Gambar 14 menunjukkan Inventory Window untuk dapat menggunakan *item* pada *inventory window* pemain dapat menekan 2 kali pada item yang ingin dipakai.

3.3.4.5. Antarmuka Loot Window



Gambar 15. Perancangan Loot Window

Pada Gambar 15 menunjukkan Loot Window yang akan berisi *item-item* yang bisa diambil oleh pemain dengan menekan pada *item* yang ingin diambil. *Window* ini akan muncul jika pemain membuka peti.

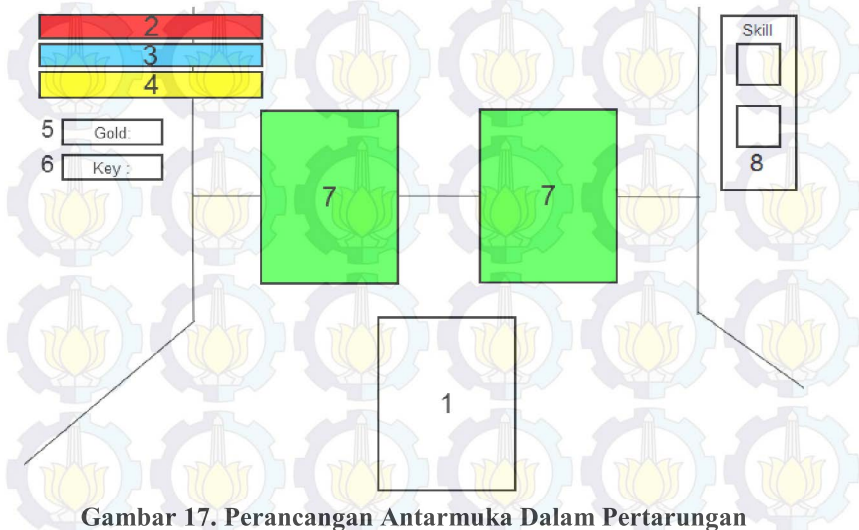
3.3.4.6. Antarmuka Shop Window



Gambar 16. Perancangan Antarmuka Shop Window

Gambar 16 menunjukkan Shop Window. *Window* ini akan muncul jika pemain menekan sebuah patung penjual yang akan ada pada permainan.

3.3.4.7. Antarmuka Dalam Pertarungan



Gambar 17. Perancangan Antarmuka Dalam Pertarungan

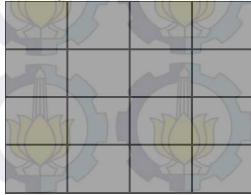
Keterangan pada Gambar 17 :

1. Posisi karakter pemain.
2. Bar darah pemain saat ini.
3. Bar mana pemain saat ini.
4. Bar exp/experient point pemain saat ini.
5. Menunjukkan *gold* pemain saat ini.
6. Menunjukkan kunci yang dimiliki pemain saat ini.
7. Posisi musuh.
8. Memberitakan *skill* apa saja yang dimiliki pemain saat ini.

Dalam permainan pemain akan mengambil *view third person*, yang artinya pemain akan bisa melihat karakternya sendiri.

3.3.5. *Maze Generator*

Dalam pembuatan *maze* tersebut menggunakan algoritma *Growing Tree*. Pertama akan dibangun sebuah kumpulan sel kubus seperti yang jumlahnya ditentukan diawal.

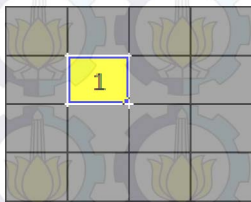


Gambar 18. Kumpulan Sel Kubus

1. Pilih secara acak satu sel, dan letakkan pada list.
2. Pilih satu sel pada list, pilih secara random tetangga sel yang belum dikunjungi, tambahkan tetangga tersebut pada list. Jika tidak ada tetangga yang bisa dikunjungi lagi hapus sel dari list.
3. Ulangi langkah 2 sampai list tersebut kosong.

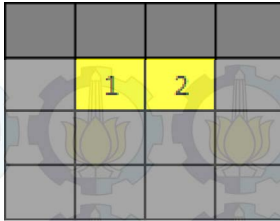
Contoh 1 pada sel kubus 4x4 jalannya algoritma *Growing Tree*:

1. Pilih satu sel dan letakkan pada list.



Gambar 19 Langkah 1 Algoritma Pada Sel Kubus 4x4

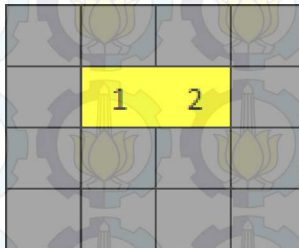
2. Pilih secara random tetangga dari sel 1,



	1	2	

Gambar 20 Langkah 2 Algoritma Pada Sel Kubus 4x4

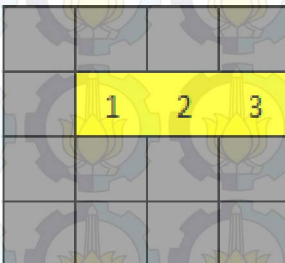
Tambahkan sel 2 pada list dan hilangkan dinding diantara sel ini.



	1	2	

Gambar 21 Langkah 3 Algoritma Pada Sel Kubus 4x4

3. Ulangi langkah 2 sampai tidak ada tetangga yang bisa dikunjungi.



	1	2	3

Gambar 22 Langkah 4 Algoritma Pada Sel Kubus 4x4

	1	2	3
			4

Gambar 23 Langkah 5 Algoritma Pada Sel Kubus 4x4

	1	2	3
			4
			5

Gambar 24 Langkah 6 Algoritma Pada Sel Kubus 4x4

	1	2	3
			4
		6	5

Gambar 25 Langkah 7 Algoritma Pada Sel Kubus 4x4

	1	2	3
			4
	7	6	5

Gambar 26 Langkah 8 Algoritma Pada Sel Kubus 4x4

	1	2	3
8			4
7	6	5	

Gambar 27 Langkah 9 Algoritma Pada Sel Kubus 4x4

	1	2	3
9	8		4
	7	6	5

Gambar 28 Langkah 10 Algoritma Pada Sel Kubus 4x4

	1	2	3
9	8		4
10	7	6	5

Gambar 29 Langkah 11 Algoritma Pada Sel Kubus 4x4

4. Karena sudah tidak ada tetangga yang bisa dikunjungi maka buang sel 10 dari list. Lalu ambil sel 9 dan lakukan langkah 3 kembali.

11	1	2	3
9	8		4
10	7	6	5

Gambar 30 Langkah 11 Algoritma Pada Sel Kubus 4x4

12			
11	1	2	3
9	8		4
10	7	6	5

Gambar 31 Langkah 12 Algoritma Pada Sel Kubus 4x4

12	13		
11	1	2	3
9	8		4
10	7	6	5

Gambar 32 Langkah 13 Algoritma Pada Sel Kubus 4x4

12	13	14	
11	1	2	3
9	8		4
10	7	6	5

Gambar 33 Langkah 14 Algoritma Pada Sel Kubus 4x4

12	13	14	15
11	1	2	3
9	8		4
10	7	6	5

Gambar 34 Langkah 15 Algoritma Pada Sel Kubus 4x4

5. Karena sel 15 tidak ada tetangga lagi, maka sel 15 dihapus dari list. Sel dihapus dari list sampai sel 8, karena sel ini masih memiliki tetangga.

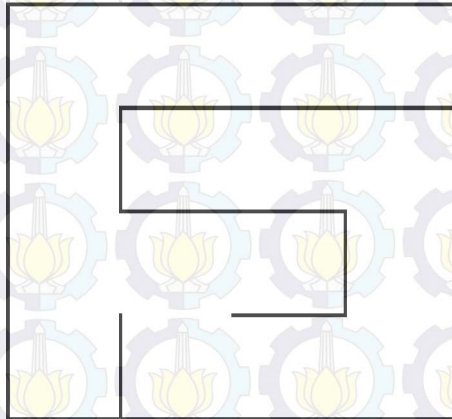
12	13	14	15
11	1	2	3
9	8		4
10	7	6	5

Gambar 35 Langkah 16 Algoritma Pada Sel Kubus 4x4

12	13	14	15
11	1	2	3
9	8	16	4
10	7	6	5

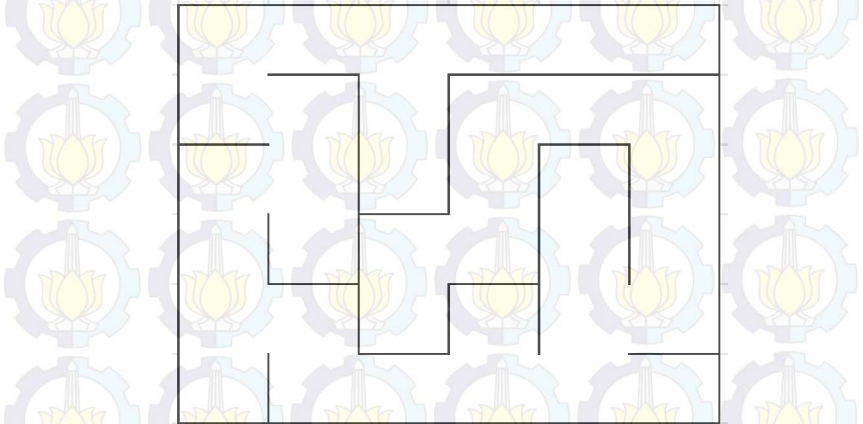
Gambar 36 Langkah 17 Algoritma Pada Sel Kubus 4x4

6. Karena sudah tidak ada tetangga yang belum dikunjungi iterasi dihentikan ditandai dengan list yang kosong.

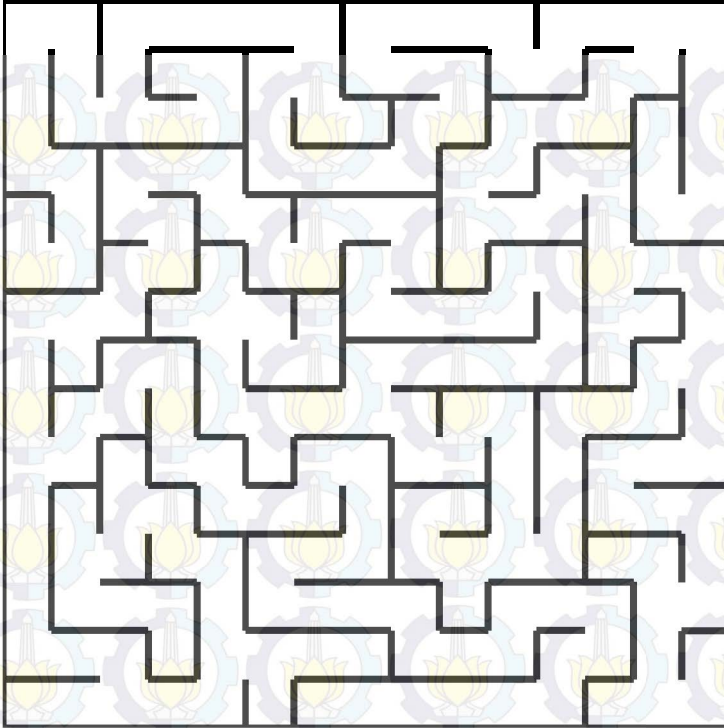


Gambar 37 Hasil Maze Dari Algoritma

Gambar 38 merupakan salah satu hasil maze dengan algoritma *Growing Tree* dengan ukuran 5x5.



Gambar 39 merupakan salah satu hasil maze dengan algoritma *Growing Tree* dengan ukuran 6x6.



Gambar 40 Maze 15x15

Gambar 40 merupakan salah satu hasil maze dengan algoritma *Growing Tree* dengan ukuran 6x6.

Setelah maze selesai dibuat akan dilakukan random untuk penempatan monster dan peti/chest dengan jumlah yang telah ditentukan pada masing-masing level.

Untuk menjamin adanya pintu keluar, akan diletakkan sebuah objek sebagai pintu keluar secara acak.

3.3.6. Level dan Skenario

Berikut rancangan level dan skenario yang akan dibangun pada tugas akhir ini, ditunjukkan pada Tabel 3.7.

Tabel 3.7. Level dan Skenario

Level	Ukuran maze (Sel x Sel)
1	6 x 6
2	7 x 7
3	8 x 8
4	9 x 9
5	10 x 10
6	16 x 16
7	17 x 17
8	18 x 18
9	19 x 19
10	20 x 20
11	26 x 26
12	27 x 27
13	28 x 28
14	29 x 29
15	30 x 30
16	36 x 36
17	37 x 37
18	38 x 38
19	39 x 39
20	40 x 40

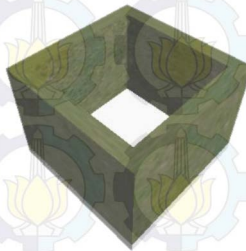
Pada Tabel 3.7 menunjukkan ukuran setiap *maze* setiap level. Pada level 1-5 pertambahan ukuran *maze* hanya satu ini dimaksudkan agar pemain bisa beradaptasi terlebih dahulu terhadap *maze*, namun untuk kelipatan 5 level ukuran *maze* ditambah 5 agar pemain lebih tertantang.

3.4. Design Asli dan Modifikasi *Growing Tree*

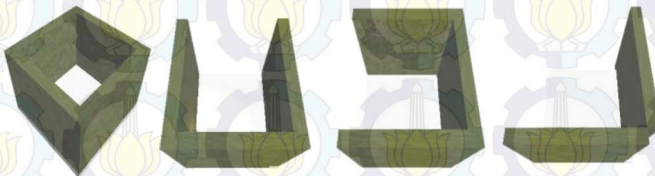
Dalam pembuatan kode untuk *Growing Tree* terdapat beberapa modifikasi baik untuk performance dalam jumlah objek ataupun menyesuaikan dengan kode di dalam permainan.

Pada design asli digunakan 1 jenis objek sebagai sel, objek tersebut ditunjukkan pada Gambar 41, potongan kode ditunjukkan pada Potongan Kode 8. Namun pada modifikasi objek yang digunakan sebagai sel ada 4 macam objek, objek ditunjukkan pada Gambar 42, sedangkan kode ditunjukkan pada Potongan Kode 2. Perubahan dilakukan karena pada design asli akan terjadi penumpukan 2 objek menjadi satu saat pembuatan kumpulan sel. Ini akan mengakibatkan jumlah objek lebih banyak dari seharusnya. Selanjutnya pada design asli pembuatan maze akan dimulai dengan suatu trigger ditunjukkan pada Potongan Kode 9, hal ini di modifikasi agar pembuatan maze dimulai tanpa memerlukan suatu trigger.

Modifikasi selanjutnya adalah pada design aslinya hasil dari maze tersebut tidak bisa disimpan, sehingga hal ini dimodifikasi agar hasil maze yang telah diciptakan bisa disimpan. Penyimpanan ini dilakukan agar maze yang dibuat tetap sama jika pemain memilih menu Load.



Gambar 41. Sel Pada Design Asli



Gambar 42. Semua Sel Pada Design Modifikasi

BAB IV IMPLEMENTASI

Bab ini membahas tentang implementasi dari perancangan sistem.

4.1. Lingkungan Implementasi

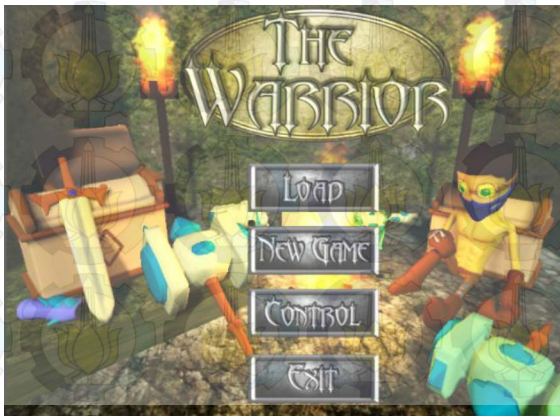
Berikut merupakan lingkungan yang digunakan saat implementasi.

Tabel 4.1. Lingkungan Implementasi

Perangkat Keras	Prosesor : Intel® CORE™ i5 CPU @ 2.53GHz Memori : 2 GB
Perangkat Lunak	Sistem Operasi : Microsoft Windows 7 64-bit Perangkat Pengembang : Unity

4.2. Implementasi Menu Utama

Pada implementasi Menu Utama terdapat 4 tombol yang bisa dipilih oleh pemain, yaitu Load, New Game, Control dan Exit. Berikut hasil implementasinya



Gambar 43. Tampilan Menu Utam

4.3. Implementasi Penggunaan Saat Permainan

Berikut merupakan *psedocode* dari beberapa FSM penggunaan saat permainan yang ada pada tahap perancangan.

```
if(Input.GetButtonDown("Attack") && !_attack &&
!_attacked){
    _forward = Forward.none;
    _attack = true;
    _animator.Play("Attack");
    if(_counterAttackAnimation == 3){
        _delayAttack = 2 * TIME_ATTACK;
    }
    _animator.SetFloat("CounterAnim",_counterAttack
Animation);
    _animator.SetBool("Attack",true);
    _speed = 0f;
    Combo();
}
```

Kode Sumber 1. Pemain menyerang

Pada Kode Sumber 1 menerapkan saat pemain menekan tombol serang/attack, pada fungsi *Combo* dilakukan penambahan *counter combo*. *Combo* ini akan berjalan apabila pemain bisa menyerang dalam kurun waktu yang telah ditentukan setelah serangan terakhir. Apabila pemain tidak bisa maka combo akan kembali menjadi 0.

```
If pemain menyerang
THEN
    If ada musuh pada jarak serang
    THEN
        Kurangi darah musuh tersebut
    ENDIF
ENDIF
```

Kode Sumber 2. Psedocode Mengurangi Darah Musuh

Kode Sumber 2 merupakan *psedocode* dari kode untuk mengurangi darah musuh. Darah musuh akan berkurang jika

pemain sedang menyerang dan musuh berada pada jarak serang pemain. Kode lengkap terlampir pada Potongan Kode 1.

```
private void DestroyThisObject() {
    if (_counterDestroy == 2f) {
        GetComponent<Animation>().Play("Die");
    }
    _counterDestroy -= Time.deltaTime;
    if (_counterDestroy <= 0) {
        GetComponent<Animation>().Stop("Die");
        Destroy(this.gameObject);
    }
}
```

Kode Sumber 3. Menghancurkan Objek Jika Mati

Kode Sumber 3 menunjukkan fungsi untuk menghancurkan objek saat mati.

```
Instantiate(gold, this.transform.position,
Quaternion.identity);
```

Kode Sumber 4. Membuat Objek Gold

Kode Sumber 4 digunakan untuk membuat objek *gold* saat musuh mati.

```
void CounterResetCombo() {
    if (_counterTimeCombo >= 0) {
        _counterTimeCombo -= Time.deltaTime;
    } else {
        _counterTimeCombo = 0;
    }
    if (_counterTimeCombo <= 0) {
        _counterAttackAnimation = 0;
    }
}
```

Kode Sumber 5. Reset Combo

Kode Sumber 5 fungsi untuk mereset *combo* saat pemain terkena serangan. *Combo* akan tereset apabila waktu yang telah ditentukan untuk pemain menyerang kembali telah habis.

```

public void DoDamage(float dmg){
    _attacked = true;
    Instantiate(bloodParticle, new
    Vector3(this.transform.position.x,
    this.transform.position.y + offsetSpawnBlood,
    this.transform.position.z),
    Quaternion.identity);
    _playerCharacter.HealthNow -= dmg;
    _animator.SetBool("Damaged",true);
    if(_playerCharacter.HealthNow <= 0){
        _animator.SetBool("Die",true);
        die = true;
        PlaySound(soundDie, false);
    }
}

```

Kode Sumber 6. Mengurangi Darah Pemain

Kode Sumber 6 digunakan untuk mengurangi darah pemain dan jika darah pemain kurang dari sama dengan 0 maka pemain mati.

```

private void DieWindow(int id){
    if(GUI.Button(new Rect(10, 30, 90, 30),
    "I'm Warrior")){
        Application.LoadLevel(PlayerPrefs.GetString("La
        st Level"));
    }
    if(GUI.Button(new Rect(110, 30, 90, 30),
    "Menu")){
        Application.LoadLevel("Menu");
    }
}

```

Kode Sumber 7. Window Saat Pemain Mati

Kode Sumber 7 digunakan untuk memunculkan *window* saat pemain mati. Jika pemain mati akan terdapat 2 pilihan, yaitu tombol *I'm warrior* dan *Menu*. Jika pemain memilih *I'm*

warrior maka sistem akan membangkitkan level terakhir pemain. Namun jika pemain memilih *Menu* maka pemain akan kembali ke menu utama.

```
void OnTriggerEnter(Collider c){
    if(c.tag == "Player"){
        GetComponent().Play();
        _showNotif = true;
        _tempAmount =
Random.Range(minAmount,maxAmount);
        _tempCounterHeight = _screenHeight -
20;

        Instantiate(goldParticle,
transform.position, Quaternion.identity);
c.gameObject.GetComponent<PlayerCharacter>().gold += _tempAmount;
this.gameObject.GetComponent<Collider>().enabled = false;

part[0].gameObject.GetComponent<SkinnedMeshRenderer>().enabled = false;
part[1].gameObject.GetComponent<MeshRenderer>().enabled = false;
        Destroy(this.gameObject,4f);
    }
}
```

Kode Sumber 8. Pemain Menyentuh *Gold*

Kode Sumber 8 saat pemain menyentuh *gold* tambahkan *gold* pemain sejumlah *gold* yang telah ditentukan dan hancurkan *gold* tersebut. *Gold* yang akan diterima oleh pemain tergantung dari nilai random yang keluar saat pertama kali *gold* dibuat. Selain itu, saat pemain mengambil *gold* tersebut akan dibuat *particle* berwarna kuning dan sebuah suara untuk menandakan bahwa pemain telah mengambil *gold* tersebut. Jadi pemain dapat mengetahui bahwa dirinya telah mengambil *gold* tersebut.


```

public void OnMouseUp() {
    if(!PlayerControl.pause){
        GameObject go =
        GameObject.FindGameObjectWithTag("Player");

        if(go == null){
            return;
        }

        if(Vector3.Distance(_myTransform.position,
        go.transform.position) > maxDistance &&
        !_inUse){

            return;
        }

        switch(state) {
            case Treasure.State.open:
                state = Treasure.State.inBetween;
                ForceClose();
                break;
            case Treasure.State.close:
                if(MyGUI.chest != null){
                    MyGUI.chest.ForceClose();
                }

                state = Treasure.State.inBetween;
                StartCoroutine("Open");
                break;
        }
    }
}

```

Kode Sumber 9. Membuka Peti

Kode Sumber 9 peti akan terbuka jika pemain memencet peti dan jarak kurang dari sama dengan jarak yang telah ditentukan. Setelah dibuka akan muncul Loot Window yang berisi item yang bisa diambil.

```

public void OnMouseUp() {
    if(!PlayerControl.pause){
        if(Vector3.Distance(transform.position,
        _player.transform.position) > maxDistance){
            return;
        }
        _animation.Play("Greeting");
        PlaySound(welcome, false);
        _playGreeting = true;
        openWindowShop = !openWindowShop;
        _openShop = true;
    }
}

```

Kode Sumber 10. *Shop*

Pada Kode Sumber 10 menunjukkan *shop* bisa dibuka saat jarak pemain dengan *shop* lebih kecil dari atau sama dengan jarak yang telah ditentukan.

```

void OnTriggerEnter(Collider c){
    if(c.tag == "Player"){
        Debug.Log("player");
        if(keyHaveNow == keyToExit){
            //save all
            _gsScripts.SaveCharacterData();
            //load next level
            int ne = _level.thisLevel + 1;
            PlayerPrefs.SetString("Last Level",
            "Level " + ne);
            string nextLevel = "Level " + ne;
            Application.LoadLevel(nextLevel);
        }
    }
}

```

Kode Sumber 11. *Exit*

Kode Sumber 11 menerapkan saat pemain memasuki pintu keluar jika kunci yang dikumpulkan sama dengan jumlah kunci yang telah ditentukan, simpan semua data terakhir dan lanjutkan ke level berikutnya.

4.4. Implementasi Kontrol

```

if(Input.GetButton("Horizontal") &&
!_attacked && !_attack ){
    if(Input.GetAxis("Horizontal") > 0){
        _turn = Turn.right;
    }else{
        _turn = Turn.left;
    }
}
if(Input.GetButton("Vertical") &&
!_attacked && !_attack){
}
if(Input.GetButtonDown("Attack") &&
!_attack && !_attacked){
}
if(Input.GetButtonUp("Toggle Inventory")){
    Messenger.Broadcast("ToggleInventory");
}
if(Input.GetButtonUp("Toggle Character
Window")){
    Messenger.Broadcast("ToggleCharacterWindow"
);
}
if(Input.GetButtonUp("Skill1") &&
_playerCharacter.allSkill[0].Available){
}
if(Input.GetButtonUp("Skill2") &&
_playerCharacter.allSkill[1].Available){
}
if(Input.GetButtonDown("Pause")){
    if(pause){
    }
    pause = !pause;
}

```

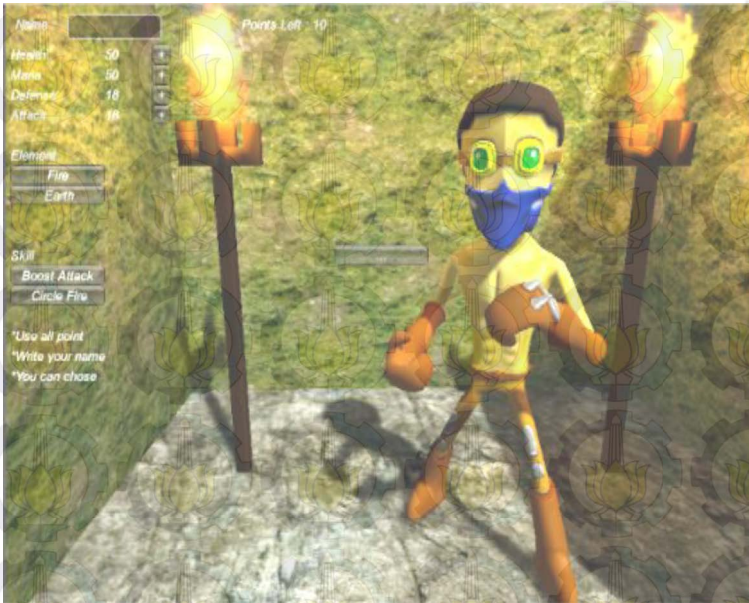
Kode Sumber 12. Kode Kontrol

Kode Sumber 12 menunjukkan untuk membuat kontrol pada permainan. Beberapa tombol terdapat syarat untuk bisa ditekan seperti *skill* terdapat syarat yaitu *skill* tersebut tidak dalam keadaan *cooldown* dan mana yang dimiliki pemain melebihi atau

sama dengan mana yang dibutuhkan untuk menggunakan *skill* tersebut.

4.5. Implementasi Antarmuka

4.5.1. Antarmuka Menu Character Creation



Gambar 44. Tampilan Character Creation

Gambar 44 merupakan hasil dari penerapan rancangan antarmuka menu Character Creation. Nama pemain bisa diisi dengan mengisi kolom nama yang terletak sebelah kiri label nama. Untuk menggunakan *point* dapat menekan tanda tambah disamping *attribute* yang ingin ditambahkan *point*.

4.5.2. Antarmuka Character Window

Pada Gambar 45 menunjukkan Character Window Tab Equipment. Terdapat 6 slot yang bisa di isi. 1 slot untuk weapon, 1 slot untuk pelindung kepala, 1 slot untuk pelindung dada, 2 slot untuk pelindung tangan dan 2 slot untuk pelindung kaki.



Gambar 45. Tampilan Character Window Tab Equipment



Gambar 46. Tampilan Tab Attributes Pada Character Window

Pada Gambar 46 menunjukkan 4 *attribute* yang dimiliki pemain, yaitu *health*, *mana*, *defense*, *attack*. Jika *point attribute* pemain sama dengan nol maka tombol pada samping *attribute* akan disembunyikan sehingga pemain tidak bisa menambahkan *point* pada *attribute* tersebut.



Gambar 47. Tampilan Point Attribute Jika Tidak Nol

Jika Point Attribute tidak nol maka akan muncul tombol tambah disamping *attribute*, seperti yang ditunjukkan pada Gambar 47. Pemain dapat menambahkan *point* ke *attribute* mana pun yang diinginkan oleh pemain. Dengan menambahkan *point* tersebut ke salah satu *attribute*, maka *attribute* tersebut akan memiliki nilai yang lebih besar dari sebelumnya.



Gambar 48. Tampilan Tab Skills Pada Character Window

Selanjutnya pada Gambar 48 menunjukkan tab Skills jika Point Skill sama dengan nol. Pemain tidak bisa menambahkan *point*.



Gambar 49. Tampilan Jika Point Skill Tidak Nol

Sedangkan Gambar 49 menunjukkan tab Skills apabila Point Skills sama dengan 1. Pemain dapat menambahkan *point* tersebut ke *skill* mana pun yang diinginkan. Dengan menambahkan *point* tersebut *skill* pemain tersebut akan memiliki efek/serangan yang lebih besar namun memerlukan mana yang lebih banyak juga.

4.5.3. Antarmuka Inventory Window



Gambar 50. Tampilan Inventory Window

Gambar 50 menunjukkan *inventory* pemain. Untuk saat ini *inventory* maksimal adalah 12 x 6. Jika *inventory* pemain penuh maka pemain dapat menghancurkan *item* yang tidak dipakai oleh pemain dengan memilih *item* tersebut lalu menekan tombol *destroy* pada pojok kanan atas *window*.

4.5.4. Antarmuka Loot Window



Gambar 51. Tampilan Loot Window

Gambar 51 merupakan Loot Window yang akan muncul apabila pemain membuka peti. Pemain dapat mengambil *item* dengan menekan *item* yang diinginkan. *Item* akan secara otomatis berpindah ke *inventory* jika *inventory* tidak penuh.

4.5.5. Antarmuka Shop Window



Gambar 52. Tampilan Shop Window

Pada Gambar 52 menunjukkan Shop Window yang mempunyai 3 tab, yaitu tab Consumable, Common, Uncommon dan Rare. Jika pemain ingin membeli maka pemain dapat menekan dua kali pada *item* yang diinginkan.

4.5.6. Antarmuka Dalam Pertarungan



Gambar 53. Tampilan Dalam Pertarungan

Gambar 53 menunjukkan hasil implementasi saat pertarungan, terdapat letak pemain, musuh, darah pemain, mana pemain, *exp* pemain saat ini, *gold* pemain saat ini, kunci yang perlu dikumpulkan pemain dan skill window yang terletak pada kanan atas layar.

4.6. Implementasi *Maze* Generator

```
public void CreateTheMaze () {
    CreateAllCell ();
    SetWhereBegin (0, 0);

    while (visitedCellList.Count != 0) {
        FindNextCell ();
    }
    PlayerPrefs.SetString ("Maze Level" +
level, _nextMaze);
}
```

Kode Sumber 13. Implementasi *Maze* Generator

Kode Sumber 13 menunjukkan fungsi utama dalam pembuatan maze. Fungsi ini diawali dengan pembuatan kumpulan sel dengan jumlah yang telah ditentukan sebelumnya, tugas ini dijalankan oleh fungsi `CreateAllCell` (kode ditunjukkan pada Potongan Kode 2 di lampiran potongan kode). Selanjutnya ditentukan dimana pembuatan maze dimulai dengan fungsi `SetWhereBegin` (kode ditunjukkan pada Potongan Kode 4 di lampiran potongan kode). Lalu dijalankan fungsi `FindNextCell` (ditunjukkan pada Potongan Kode 3 di lampiran potongan kode) sampai list pada `visitedCellList` habis. Terakhir simpan data untuk membuat *maze* yang sama.

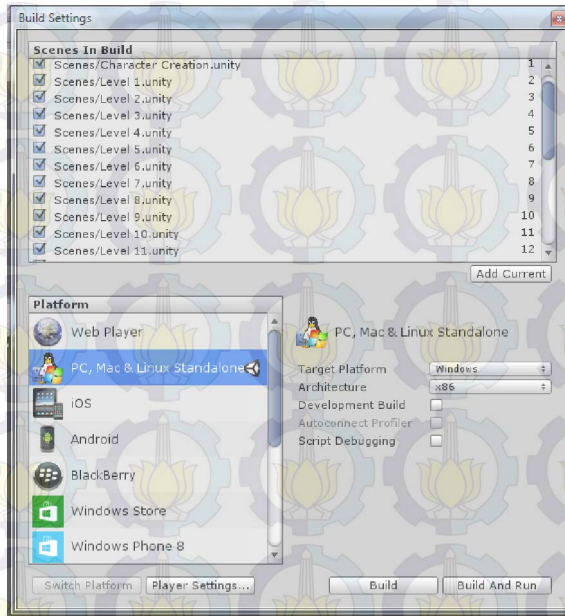
```
private void SpawnExit(){
    bool done = false;
    do{
        for(int i = 0; i <
            _allPlaceAvailable.Count; i++){
            int rn = Random.Range(0,3);
            if(rn == 1){
                Instantiate(exitDoor, new
                Vector3((_allPlaceAvailable[i].x *
                _allcel.buffer), 0f, (_allPlaceAvailable[i].z *
                _allcel.buffer)), Quaternion.identity);
                done = true;
            }
            return;
        }
    }while(!done);
}
```

Kode Sumber 14. Membuat Pintu Keluar

Pintu keluar dibangun secara acak di dalam sel pada maze yang telah dihubungkan dengan minimal satu sel tetangganya, karena hal tersebut pemain dapat dipastikan bisa mencapai pintu keluar. Pada penempatan pintu keluar sudah dapat dipastikan saat meletakkan pintu keluar tidak akan sama dengan penempatan peti. Sehingga untuk sebelum mencapai pintu keluar pemain

harus mencari peti dengan kunci yang tersembunyi di dalamnya. Jika belum bisa menemukan kunci tersebut pemain tidak akan bisa keluar dari maze dan melanjutkan ke maze berikutnya dengan ukuran yang lebih besar.

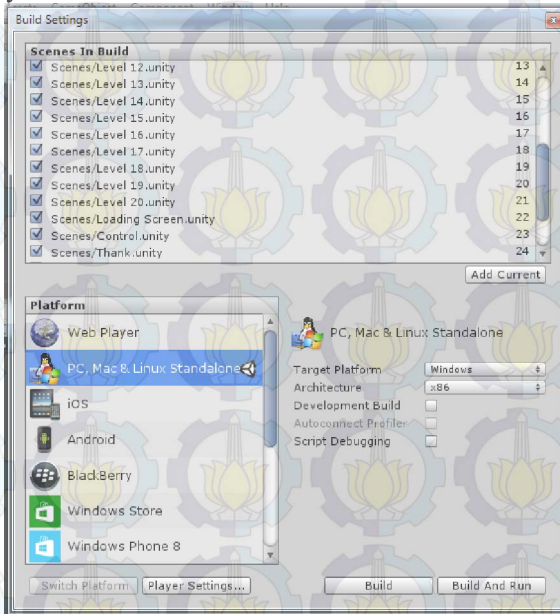
4.7. Implementasi Level dan Skenario



Gambar 54. Screenshoot dari Unity Untuk Level 1-11

Gambar 54 merupakan pengaturan untuk level 1 sampai level 11 di dalam unity. Pada setiap *scene* yang dimasukkan dalam pengaturan level telah diatur semua ukuran *maze* sesuai dengan skenario yang telah dibuat. Pada *level 1-5* penambahan ukuran sisi *maze* hanya satu. Ini dimaksudkan agar pemain bisa beradaptasi dengan permainan. Level 6 memiliki perbedaan 5 pada ukuran sisi dengan level 5. Begitu juga dengan level kelipatan 5 berikutnya akan memiliki perbedaan ukuran sisi sebanyak 5.

Gambar 55 implementasi level 12 sampai dengan level 20 pada unity.



Gambar 55. Implementasi Level 12-20

4.8. Pembuatan Karakter

Pembuatan semua model 3D pada *video game* ini menggunakan perangkat lunak Blender. Sedangkan untuk pengerjaan texturnya menggunakan GIMP. Model terbagi menjadi 2 kelompok. Kelompok pertama model 3D yang terdapat animasinya dan model 3D tanpa animasi.

Model 3D yang menggunakan animasi antara lain model untuk karakter utama, monster, dan peti. Sedangkan model 3D yang tidak terdapat animasinya adalah model untuk senjata, gold, pelindung dada, tangan, kaki dan kepala. Untuk item yang bisa di konsumsi tidak dibuat model 3Dnya dikarenakan item tersebut hanya akan digunakan texture 2D saja pada Inventory Window atau Loot Window.



Gambar 56. Karakter Utama

Gambar 56 merupakan karakter utama yang akan dimainkan oleh pemain pada permainan. Beberapa animation yang ada pada karakter ini antara lain berjalan, menyerang, dan menggunakan skill.



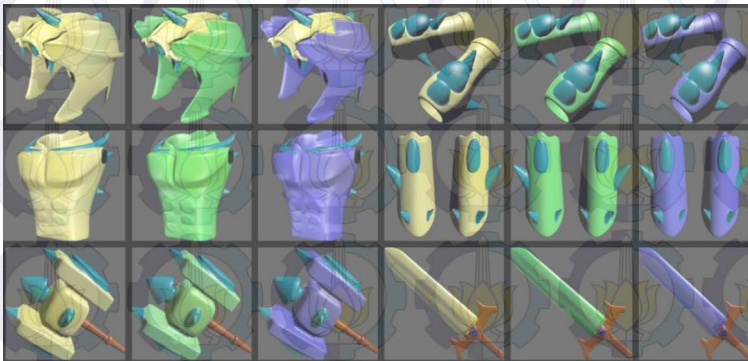
Gambar 57. Peti

Gambar 57 Merupakan model untuk peti yang bisa dibuka oleh pemain. Model ini memiliki animation membuka dan menutup peti. Jika peti ini dibuka maka akan muncul Loot Window, di dalam Loot Window tersebut terdapat item yang bisa diambil.



Gambar 58. Monster

Gambar 58 merupakan 3 model 3D untuk monster yang digunakan di dalam permainan. Animasi yang terdapat pada karakter adalah berjalan dan menyerang.



Gambar 59. All Item

Gambar 59 merupakan semua item yang bisa digunakan di dalam permainan. Terdapat 3 tingkatan item yaitu *common*, *uncommon* dan *rare* pada setiap jenis itemnya.

BAB V

PENGUJIAN DAN EVALUASI

Bab ini akan dijelaskan rangkaian uji coba dan evaluasi terhadap tugas akhir ini. Proses pengujian dilakukan menggunakan metode *blackbox* berdasarkan skenario yang telah ditentukan.

5.1. Lingkungan Uji Coba

Lingkungan pengujian sistem pada pengerjaan tugas akhir ini dilakukan pada lingkungan dan alat kaku sebagai berikut:

Tabel 5.1. Lingkungan Uji Coba

Perangkat Keras	Prosesor : Intel® CORE™ i5 CPU @ 2.53GHz Memori : 2 GB
Perangkat Lunak	Sistem Operasi : Microsoft Windows 7 64-bit Perangkat Pengembang : Unity

Di dalam dokumentasi Unity disebutkan OS yang bisa menjalankan hasil build dari Unity adalah Windows XP+, Mac OS X 10.7+, Ubuntu 12.04+, dan SteamOS+.

5.2. Skenario Uji Coba

Tabel 5.2 menunjukkan beberapa skenario yang akan diuji cobakan terhadap perangkat lunak yang telah dibuat. Terdapat 12 skenario yang akan diuji coba. Dua belas skenario dipilih karena dianggap bisa mempresentasikan fungsional perangkat lunak. Pengujian dilakukan dengan menjalankan perangkat lunak lalu mencoba menjalankan perangkat lunak sesuai dengan skenario yang telah dibuat.

Untuk pengujian pembuatan *maze* dibuat *scene* sendiri untuk pengujian dengan kamera yang ditaruh diatas *maze*, dikarenakan jika pengujian pada *scene* level menggunakan

kamera yang mengikuti karakter maka hasil dari maze tidak akan terlihat.

Tabel 5.2. Tabel Skenario Uji Coba

No	Nama	Keterangan
1	Menggunakan <i>Combo</i>	Pemain menyerang musuh sehingga <i>combo</i> keluar
2	Mengurangi Darah Musuh	Musuh menerima serangan pemain sehingga darah musuh berkurang dan menjatuhkan gold apabila darah musuh kurang dari atau sama dengan 0.
3	Mendapatkan <i>Gold</i>	Pemain melewati <i>gold</i> sehingga jumlah <i>gold</i> pemain bertambah
4	Pemain Mati	Pemain menerima serangan hingga darah pemain kurang dari 0 dan memunculkan <i>window</i> pemain mati
5	<i>Pause</i>	Pemain menekan tombol P untuk pause pada permainan
6	Memunculkan Character Window	Pemain menekan tombol C untuk memunculkan Character Window
7	Memunculkan Inventory Window	Pemain menekan tombol I untuk memunculkan Inventory Window
8	Membuka Peti	Pemain membuka peti
9	Mengambil Item	Pemain mengambil <i>item</i> dari Loot Window
10	Membeli Item	Pemain membuka <i>shop</i> lalu membeli <i>item</i> yang diinginkan
11	Pembuatan Maze	Mencoba membangun beberapa <i>maze</i> dalam beberapa kali coba untuk melihat apakah <i>maze</i> yang dibangun sudah berbeda
12	Peletakan <i>Exit</i>	Menunjukkan bahwa tempat <i>exit</i> bisa dicapai oleh pemain

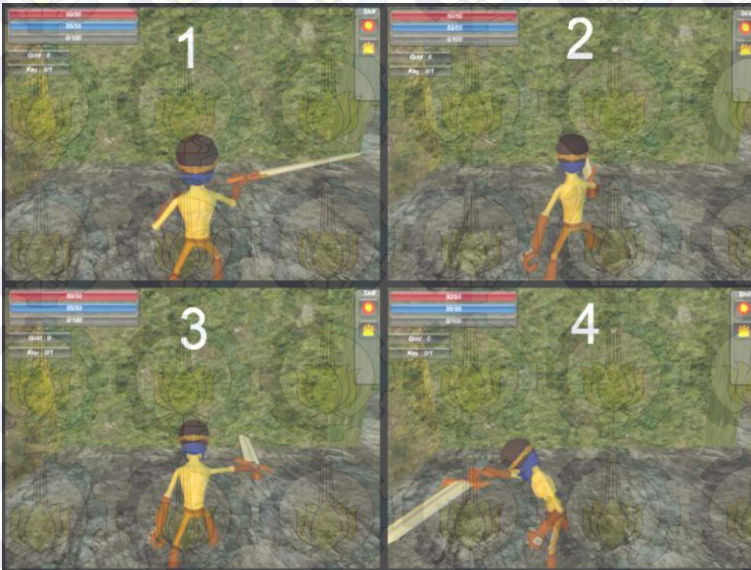
5.3. Hasil Uji Coba

Kondisi awal dari semua uji coba adalah pemain telah membuat karakter dan masuk dalam *maze*.

5.3.1. Menggunakan *Combo*

Pemain memiliki 4 serangan berbeda sebagai *combo*. Setiap kali pemain menyerang, *counter* untuk *combo* ini bertambah. Jika pemain terkena serangan atau *counter combo* sudah mencapai 4 maka *counter combo* menjadi 0. Berikut 4 serangan *combo* pada Gambar 60.

Pada serangan no 1 merupakan gerakan menebas dari kanan ke depan. Lalu pada serangan no 2 merupakan gerakan menusuk ke depan. Serangan no 3 seperti gerakan pada serangan no 1 namun gerakan no 3 menebas dari kir ke depan. Terakhir serangan no 4 merupakan gerakan menebas juga namun jangkuaangan serangan ini lebih lebar dari serangan no 1 dan 3.



Gambar 60. Semua Serangan *Combo*

5.3.2. Mengurangi Darah Musuh

Darah musuh akan berkurang jika terkena serangan biasa dari pemain atau skill dari pemain. Darah musuh berbentuk kubus berwarna yang berada di atas musuh. Jika kubus habis berarti darah musuh telah 0 ditunjukkan pada Gambar 61.

Jumlah darah yang berkurang merupakan jumlah penjumlahan *attack* pada karakter ditambah dengan jumlah *attack* yang terdapat pada senjata. Setiap senjata memiliki nilai *attack* minimal dan nilai *attack* maksimal. Setiap menyerang *attack* yang akan dikeluarkan senjata akan diacak antara nilai *attack* minimal dan nilai *attack* maksimal.



Gambar 61. Darah Musuh Berkurang

5.3.3. Mendapatkan Gold

Setelah membunuh musuh, maka akan terdapat gold yang dijatuhkan oleh musuh. Jumlah gold yang dijatuhkan diacak diantara dua nilai yang telah ditentukan.

Pada Gambar 62 jumlah awal gold pemain 0. Setelah menyentuh objek gold yang terdapat di atas tanah maka gold pemain bertambah 26 ditunjukkan pada Gambar 63.



Gambar 62. Gold Pemain 0



Gambar 63. Gold Pemain Bertambah

5.3.4. Pemain Mati

Saat darah pemain 0, maka akan muncul *die window*. Di dalam *window* ini terdapat pilihan untuk *respawn* kembali pada *maze* tersebut atau kembali ke menu utama. *Window* tersebut ditunjukkan pada Gambar 64. Di dalam *window* terdapat 2 tombol yaitu tombol I'm warrior untuk mengulang permainan pada level terakhir dan tombol Menu untuk kembali ke menu utama.



Gambar 64. Pemain Mati

5.3.5. *Pause*

Gambar 65 menunjukkan jika pemain menekan tombol P. Setelah menekan tombol *pause* ini, maka permainan akan berhenti untuk sementara. Di dalam *window pause* ini terdapat tombol Back to Main Menu untuk kembali ke menu utama permainan. Selanjutnya tombol Kontrol untuk melihat tombol apa saja yang bisa dipakai selama permainan. Untuk *unpause* permainan pemain dapat menekan tombol P kembali.



Gambar 65. Pause

5.3.6. Memunculkan Character Window

Gambar 66 merupakan tampilan ketika pemain menekan tombol C. Terdapat 3 tab yang bisa dipilih Equipment untuk melihat item apa saja yang dipakai oleh pemain. Tab Attribute digunakan untuk melihat *attribute* dan menambahkan *point* jika memiliki Point Attribute. Terakhir tab Skills untuk melihat *skill* dan menambahkan *point* jika mempunyai *point*. Window ini bisa digeser sesuai keinginan pemain.



Gambar 66. Character Window

5.3.7. Memunculkan Inventory Window

Gambar 67 merupakan tampilan ketika pemain menekan tombol I. Inventory Window ini bisa digeser dan ditempatkan sesuai keinginan pemain. Untuk menutupnya bisa menekan tombol I kembali atau menekan tombol silang pada pojok kiri atas *window*.



Gambar 67. Inventory Window

5.3.8. Membuka Peti

Gambar 68 menunjukkan sebelum dan sesudah peti dibuka. Setelah membuka peti akan terbual Loot Window. Pada *window* ini akan terdapat item yang bisa diambil oleh pemain. Loot Window ini akan secara otomatis tertutup jika pemain berjalan menjauhi peti atau *item* yang terdapat pada *window* ini telah diambil semua. Selain itu pemain juga bisa menutup *window* ini secara manual dengan menekan tombol silang pada pojok kiri atas Loot Window. *Window* ini tidak bisa digeser seperti 2 *window* sebelumnya.



Gambar 68. Membuka Peti

5.3.9. Mengambil *Item*

Gambar 69 menunjukkan setelah membuka peti maka Loot Window akan muncul. Pemain bisa memilih mengambil *item* dengan menekan *item* yang diinginkan. Jika pemain mengambil semua *item* yang terdapat pada Loot Window maka *window* tersebut akan tertutup secara otomatis dan peti akan dihancurkan.

Gambar 69. Mengambil *Item* dari Loot Window

5.3.10. Membeli *Item*

Sebelum dapat membeli *item* pemain harus menekan penjual dan berada pada jarak yang cukup dekat untuk membuka Shop Window. Setelah membuka Shop Window pemain dapat membeli *item* yang diinginkan dengan *double left click* pada *item* tersebut ditunjukkan pada Gambar 70.



Gambar 70. Membeli Item

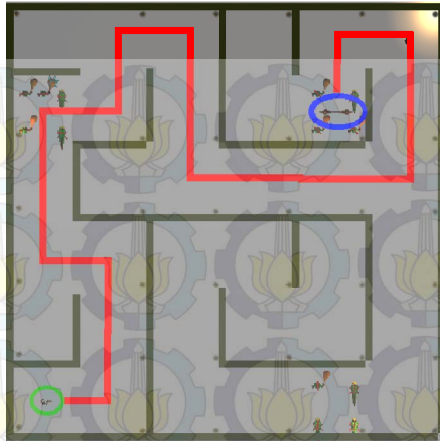
5.3.11. Pembuatan *Maze*

Berikut merupakan *maze* yang dihasilkan dari *maze* generator pada perangkat lunak ini. Setiap ukuran *maze* dicoba sebanyak 3 kali. Dan dihasilkan 3 *maze* dengan ukuran yang sama namun berbeda. Lingkaran hijau merupakan pemain dan lingkaran biru pintu keluar dari *maze* untuk melanjutkan ke *maze* berikutnya. Garis merah merupakan jalan dari pemain menuju pintu keluar.

Gambar 71 merupakan hasil pertama dari *maze* generator dengan ukuran 6 x 6.

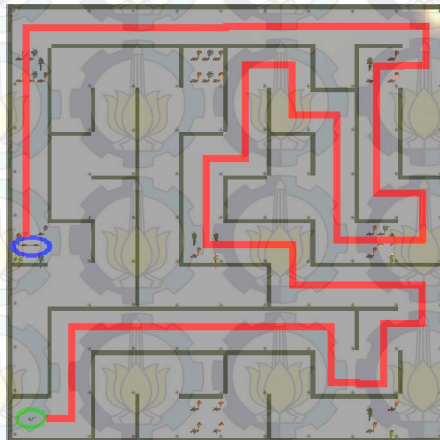
Gambar 72 merupakan hasil kedua dari *maze* generator dengan ukuran 6 x 6.

Gambar 72 merupakan hasil kedua dari *maze* generator dengan ukuran 6 x 6.



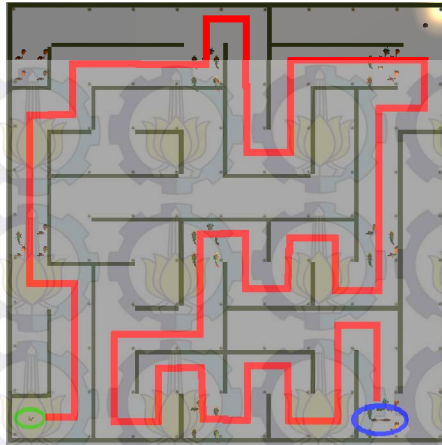
Gambar 73. Hasil 3 Maze 6x6

Gambar 73 merupakan hasil ketiga dari *maze* generator dengan ukuran 6 x 6.



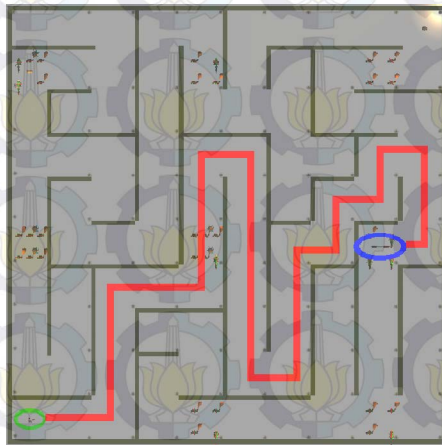
Gambar 74. Hasil 1 Maze 10x10

Gambar 74 merupakan hasil pertama dari *maze* generator dengan ukuran 10 x 10.



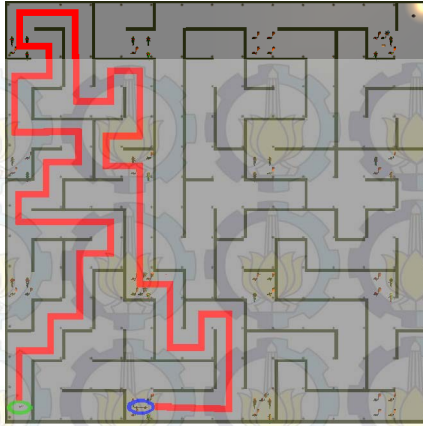
Gambar 75. Hasil 2 Maze 10x10

Gambar 75 merupakan hasil kedua dari *maze* generator dengan ukuran 10 x 10.



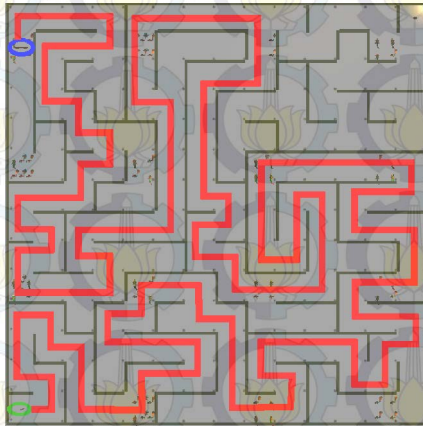
Gambar 76. Hasil 3 Maze 10x10

Gambar 76 merupakan hasil ketiga dari *maze* generator dengan ukuran 10 x 10.



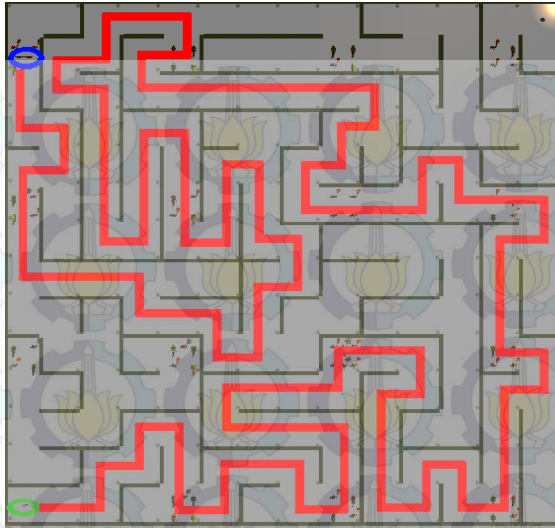
Gambar 77. Hasil 1 Maze 14x14

Gambar 77 merupakan hasil pertama dari *maze* generator dengan ukuran 14 x 14.



Gambar 78. Hasil 2 Maze 14x14

Gambar 78 merupakan hasil kedua dari *maze* generator dengan ukuran 14 x 14.



Gambar 79. Hasil 3 Maze 14x14

Gambar 79 merupakan hasil ketiga dari *maze* generator dengan ukuran 14 x 14.

5.4. Evaluasi

Dari uji coba yang telah dilakukan dapat disimpulkan bahwa semua skenario telah berjalan seperti yang diinginkan. Pada pembuatan *maze* dapat dilihat bahwa *maze* yang dibuat berbeda dan pintu keluar pasti bisa dicapai oleh pemain.

5.5. Kuisisioner

Dalam pengujian permainan ini diadakan suatu kuisisioner dengan range nilai 1-4, dengan ketentuan semakin besar nilai semakin baik penilaian. Terdapat 3 hal yang ditanyakan, yaitu mengenai antarmuka, level pada *maze* dan *fun*. Pada Tabel 5.3 ditunjukkan rata-rata hasil dari kuisisioner. Sedangkan kuisisioner dapat dilihat pada bagian lampiran kuisisioner. Kuisisioner diberikan kepada 5 orang yang telah mencoba perangkat lunak.

Tabel 5.3. Hasil Kuisioner

No.	Pernyataan	Rata –rata Penilaian
Penilaian Antarmuka		
1	Keindahan tampilan aplikasi	3,4
2	Antarmuka mudah untuk dioperasikan	3,6
Penilaian Level		
1	Level memberikan waktu adaptasi	3,2
2	Semakin besar level semakin menantang	3
Penilaian Fun		
1	Saya memiliki ketertarikan untuk bermain	3,4
2	Saya menikmati permainan	3,2
3	Saya ingin mencoba lagi permainan ini	3

Dari Tabel 5.3 dapat dilihat bahwa antarmuka sudah dapat diterima dengan baik oleh pemain. Lalu pembuatan level juga sudah bisa membuat pemain tertantang. Respon pemain untuk kategori *fun* sudah cukup bagus.

BAB VI

KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil selama proses perancangan, implementasi, serta pengujian dapat diambil kesimpulan sebagai berikut:

1. Perangkat lunak sudah dapat menerapkan skenario, level dan *gameplay roleplaying game* yang telah dirancang sebelumnya.
2. *Maze generator* pada perangkat lunak sudah dapat membuat maze yang berbeda dengan ukuran yang sama dan pintu keluar di setiap maze pasti bisa dicapai oleh pemain.
3. Antarmuka di dalam permainan sudah dapat diterima dengan baik oleh pemain, pembuatan level juga sudah bisa membuat pemain tertantang dan respon pemain untuk kategori fun sudah cukup bagus.

6.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang yaitu menambahkan cerita pada perangkat lunak sehingga perangkat lunak lebih menarik.

DAFTAR PUSTAKA

- [1] N. Oxford, "RPG/Role-Playing Game," About, [Online]. Available: <http://ds.about.com/od/glossary/g/Rpg-Role-Playing-Game.htm>. [Diakses 27 May 2015].
- [2] E. Jo, "Introduction to Mazes and Labyrinths," Edkins Jo, [Online]. Available: <http://gwydir.demon.co.uk/jo/maze/intro/>. [Diakses 29 May 2015].
- [3] C. Corporation, "What Is a Maze?," Conjecture Corporation, [Online]. Available: <http://www.wisegeek.com/what-is-a-maze.htm>. [Diakses 29 May 2015].
- [4] "The best development platform for creating games," Unity, [Online]. Available: <https://unity3d.com/unity>. [Diakses 29 May 2015].
- [5] Blender, "About," Blender, [Online]. Available: <http://www.blender.org/about/>. [Diakses 29 May 2015].
- [6] Jamis, "Maze Generation: Growing Tree algorithm," Jamis, [Online]. Available: <http://weblog.jamisbuck.org/2011/1/27/maze-generation-growing-tree-algorithm>. [Diakses 29 May 2015].

BAB VI

KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil selama proses perancangan, implementasi, serta pengujian dapat diambil kesimpulan sebagai berikut:

1. Perangkat lunak sudah dapat menerapkan skenario, level dan *gameplay roleplaying game* yang telah dirancang sebelumnya.
2. *Maze generator* pada perangkat lunak sudah dapat membuat maze yang berbeda dengan ukuran yang sama dan pintu keluar di setiap maze pasti bisa dicapai oleh pemain.
3. Antarmuka di dalam permainan sudah dapat diterima dengan baik oleh pemain, pembuatan level juga sudah bisa membuat pemain tertantang dan respon pemain untuk kategori fun sudah cukup bagus.

6.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang yaitu menambahkan cerita pada perangkat lunak sehingga perangkat lunak lebih menarik.

LAMPIRAN POTONGAN KODE

```

if(_attack){
    _counterTimeAttack -= Time.deltaTime;
    if(_counterTimeAttack <= _delayAttack &&
    !_alreadyDamaged){
        if(allEnemyInRange.Count > 0){
            for(int i = 0; i <
allEnemyInRange.Count; i++){
                float dist = Vector3.Distance(
allEnemyInRange[i].transform.position,
_myTransform.position);
                if(_counterAttackAnimation >=
3){
                    if(dist <=
_playerCharacter.DistanceAttack * 2){
                        if(allEnemyInRange[i].StateEnemyNow !=
SimpleAI.StateEnemy.Attacked){
                            Weapon weaponNow =
PlayerCharacter.EquipWeapon.GetComponent<Weapon
>();
                            float tempDmg =
Random.Range(weaponNow.minAttack,
weaponNow.maxAttack);
                            tempDmg +=
_playerCharacter.allAttribute[3].Value;
                            allEnemyInRange[i].DoDamage(tempDmg);
                        }
                    }else{
                        if(dist <=
_playerCharacter.DistanceAttack){
                            Vector3 dir2 =
(allEnemyInRange[i].transform.position -
_myTransform.position).normalized;
                            float direction2 =

```

```

Vector3.Dot (dir2, _myTransform.forward);
                                if(direction2 >
_playerCharacter.MaxAngleAttack &&
allEnemyInRange[i].StateEnemyNow !=
SimpleAI.StateEnemy.Attacked){
                                Weapon weaponNow =
PlayerCharacter.EquipWeapon.GetComponent<Weapon
>();
                                float tempDmg =
Random.Range (weaponNow.minAttack,
weaponNow.maxAttack);
                                tempDmg +=
_playerCharacter.allAttribute[3].Value;
allEnemyInRange[i].DoDamage (tempDmg);
                                }
                                }
                                }
                                _alreadyDamaged = true;
                                }else{
                                _alreadyDamaged = true;
                                }
                                }else if(_counterTimeAttack <= 0){
                                _counterTimeAttack = TIME_ATTACK;
                                _attack = false;
                                _animator.SetBool ("Attack",false);
                                _delayAttack = TIME_ATTACK * 0.5f;
                                _alreadyDamaged = false;
                                }
                                }

```

Potongan Kode 1. Kode Menyerang

```

public void CreateAllCell(){
    Transform _newCellTemp;
    _allCell = new Transform[sizeX, sizeZ];
    for (int x = 0; x < sizeX; x++) {
        for (int z = 0; z < sizeZ; z++) {
            if((x == 0) && (z != (sizeZ - 1))){
                _newCellTemp = Instantiate
                (cellPrefeb2, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }else if((x == 0) && (z == (sizeZ -
1))) {
                _newCellTemp = Instantiate
                (cellPrefeb1, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }else if((z == sizeZ - 1)){
                _newCellTemp = Instantiate
                (cellPrefeb3, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }else{
                _newCellTemp = Instantiate
                (cellPrefeb4, new Vector3 (x, 0, z) * buffer,
                Quaternion.identity) as Transform;
            }
            _newCellTemp.name = string.Format
            ("({0},0,{1})", x, z);
            _newCellTemp.parent = transform;
            _newCellTemp.GetComponent<Cell>().Position =
            new Vector3 (x, 0, z);
            _allCell[x, z] = _newCellTemp;
        }
    }
}

```

Potongan Kode 2. Fungsi CreateAllCell


```

private void FindNextCell() {
    Transform _nextCell;
    Cell _nextCellScript;

    int _nextCellPosX;
    int _nextCellPosZ;

    Transform _nowCell = visitedCellList[0];
    Cell _nowCellScript =
        visitedCellList[0].GetComponent<Cell> ();

    int _nowCellPosX =
        (int)_nowCellScript.Position.x;
    int _nowCellPosZ =
        (int)_nowCellScript.Position.z;

    int _randNumber = Random.Range(0, 4);
    int _counter = 0;
    Vector3 _randDirection;

    do{
        do{
            _randDirection =
                directions[_randNumber];
            _nextCellPosX = _nowCellPosX +
                (int)_randDirection.x;
            _nextCellPosZ = _nowCellPosZ +
                (int)_randDirection.z;
            _randNumber = (_randNumber + 1) % 4;
            _counter++;
            if (_counter > 4) {
                _nextMaze = _nextMaze + "-" +
                    "#";
                RemoveCompletedCellList(_nowCell);
                return;
            }
        }while(_nextCellPosX < 0 ||
            _nextCellPosZ < 0 || _nextCellPosX >= sizeX ||
            _nextCellPosZ >= sizeZ);
    }
}

```

```

        _nextCell = _allCell[_nextCellPosX,
        _nextCellPosZ];
        _nextCellScript = _nextCell.GetComponent
        <Cell> ();
        }while(_nextCellScript.AlreadyVisited);

        _nextMaze = _nextMaze + _nextCellPosX + ", "
        + _nextCellPosZ + "#";

        DestroyWalls(_nowCell, _nextCell);

        AddVisitedCellList(_nextCell);
    }

```

Potongan Kode 3. Fungsi FindNextCell

```

private void SetWhereBegin(int a, int b){
    AddVisitedCellList(_allCell[a, b]);
}

```

Potongan Kode 4. Fungsi SetWhereBegin

```

private void AddVisitedCellList(Transform
visitedCell){
    visitedCellList.Insert (0, visitedCell);

    Cell cs = visitedCell.GetComponent<Cell> ();
    cs.AlreadyVisited = true;
}

```

Potongan Kode 5. Fungsi AddVisitedCellList

```

private void RemoveCompletedCellList(Transform
cm){
    visitedCellList.Remove(cm);
}

```

Potongan Kode 6. Fungsi RemoveCompletedCellList

```

private void DestroyWalls(Transform a, Transform
b) {
    RaycastHit[] hits;

    Vector3 or = new Vector3(a.position.x,
a.position.y + heightRayCast, a.position.z);

    hits = Physics.RaycastAll(or, b.position -
a.position, longRayCast);

    Destroy(hits[0].transform.gameObject);
}

```

Potongan Kode 7. Fungsi DestroyWalls

```

public void CreateGrid ()
{
    int x = (int)GridSize.x;
    int z = (int)GridSize.z;
    int maxXZ = Mathf.Max (x, z);
    Camera.mainCamera.transform.position =
new Vector3 (maxXZ / 2f, maxXZ, maxXZ / 8f);
    GridArr = new Transform[x, z];
    Transform newCell;
    for (int ix = 0; ix < x; ix++) {
        for (int iz = 0; iz < z; iz++) {
            newCell = (Transform)Instantiate
(CellPrefab, new Vector3 (ix, 0, iz) * Buffer,
Quaternion.identity);
            newCell.name = string.Format
("{0},{1}", ix, iz);
            newCell.parent = transform;
            newCell.GetComponent<Cell>
().Position = new Vector3 (ix, 0, iz);
            GridArr [ix, iz] = newCell;
        }
    }
}

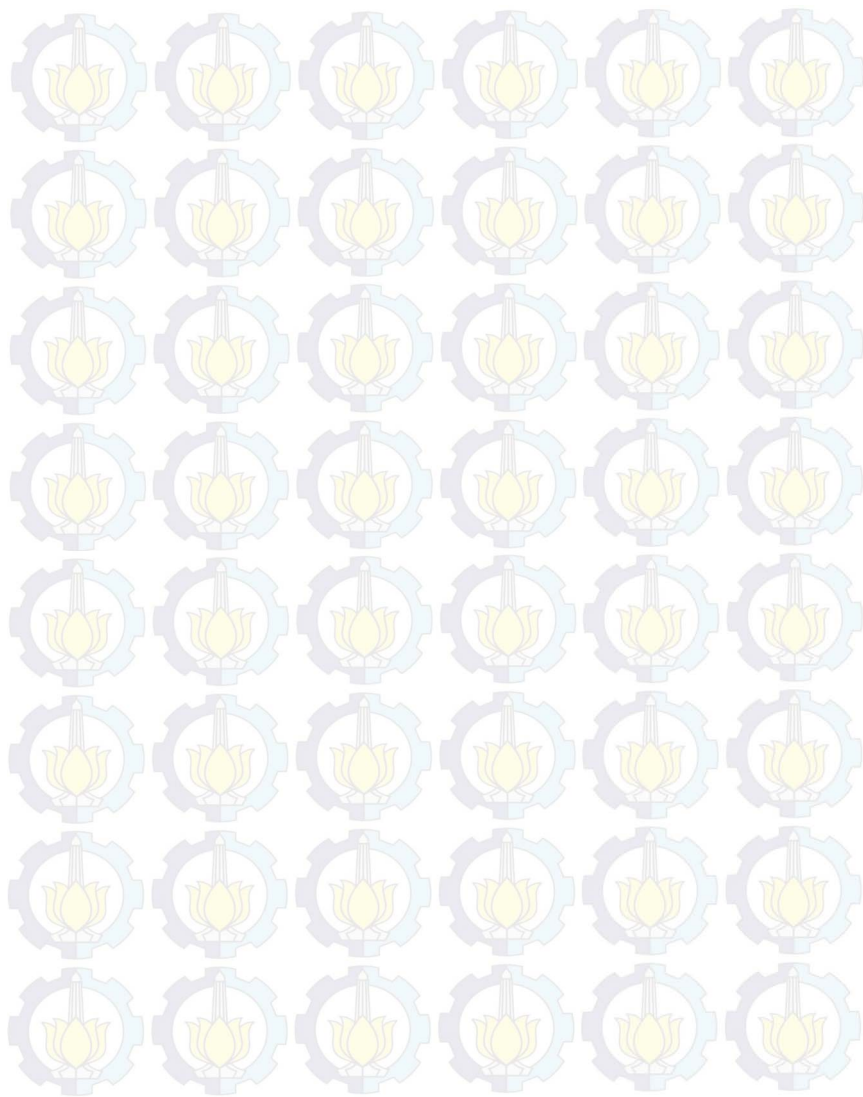
```

Potongan Kode 8. Potongan Kode Sel Dengan Satu Object

```
void Update ()
{
    if (Input.GetKeyDown (KeyCode.Tab) ||
    Input.GetKey (KeyCode.Space)) {
        InvokeRepeating ("FindNext", 0,
        0.001f); //FindNext ();
    }
    if (Input.GetKeyDown
    (KeyCode.LeftShift)) {
        FindNext ();
    }
    if (Input.GetKeyDown (KeyCode.F1))
    {
        Application.LoadLevel (0);
    }
}
```

Potongan Kode 9. Pembuatan *Maze* Dengan Triger

[Halaman ini sengaja dikosongkan]



LAMPIRAN KUISIONER


KUISIONER THE WARRIOR

Nama Lengkap	M. Labat Kustamadi
Jenis Kelamin	Perempuan / Laki-Laki
Usia	22 tahun
Pekerjaan	Mahasiswa
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '√' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi				✓
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi				✓
2	Semakin besar level semakin menantang				✓
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain				✓
2	Saya menikmati permainan				✓
3	Saya ingin mencoba lagi permainan ini			✓	

Sarabaya, 25 Juni 2015


(-----)
M. LABAT K.

KUISIONER THE WARRIOR

Nama Lengkap	MARAUU TOTO NEGORO
Jenis Kelamin	Pemuaan / Laki-Laki
Usia	21 tahun
Pekerjaan	MAHASISWA
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi			✓	
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi			✓	
2	Semakin besar level semakin menantang			✓	
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain				✓
2	Saya menikmati permainan			✓	
3	Saya ingin mencoba lagi permainan ini				✓

Surabaya, 25 Juni 2015

(MARAUU TOTO NEGORO)

KUISIONER THE WARRIOR

Nama Lengkap	Dzik Perwanto
Jenis Kelamin	Pemampuan / Laki-Laki
Usia	22 tahun
Pekerjaan	Mahasiswa
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi				✓
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi				✓
2	Semakin besar level semakin menantang				✓
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain				✓
2	Saya menikmati permainan				✓
3	Saya ingin mencoba lagi permainan ini				✓

Surabaya, 25 Juni 2015

Dzik Perwanto
(Dzik Perwanto)


KUISIONER THE WARRIOR

Nama Lengkap	Luthfan Aqsa Peradnan
Jenis Kelamin	Perempuan / Laki-Laki
Usia	21 tahun
Pekerjaan	Mahasiswa
Instansi	ITS

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi			✓	
2	Antarmuka mudah untuk dioperasikan				✓
Penilaian Level					
1	Level memberikan waktu adaptasi				✓
2	Semakin besar level semakin menantang				✓
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain			✓	
2	Saya menikmati permainan			✓	
3	Saya ingin mencoba lagi permainan ini				✓

Surabaya, 24 Juni 2015


(Luthfan Aqsa Peradnan)

KUISIONER THE WARRIOR

Nama Lengkap	Punggi Esthi Bawono
Jenis Kelamin	Perempuan / Laki-Laki
Usia	22 tahun
Pekerjaan	Mahasiswa
Instansi	Institut Teknologi Sepuluh Nopember

Setelah memainkan aplikasi tolong berikan nilai kesetujuan 1 - 4 pada tiap poin pernyataan. Semakin besar nilai yang Anda berikan menunjukkan semakin baik aplikasi permainan simulasi ini. Berikan tanda '✓' pada kolom nilai yang Anda pilih.

No.	Pernyataan	Penilaian			
		1	2	3	4
Penilaian Antarmuka					
1	Keindahan tampilan aplikasi				✓
2	Antarmuka mudah untuk dioperasikan			✓	
Penilaian Level					
1	Level memberikan waktu adaptasi			✓	
2	Semakin besar level semakin menantang		✓		
Penilaian Fun					
1	Saya memiliki ketertarikan untuk bermain			✓	
2	Saya menikmati permainan			✓	
3	Saya ingin mencoba lagi permainan ini			✓	

Surabaya, 25 Juni 2015

Punggi E.B
(-----)